
AFRL-RY-WP-TR-2016-0127

# MATHEMATICS OF SENSING, EXPLOITATION, AND EXECUTION (MSEE)

## Sensing, Exploitation, and Execution (SEE) on a Foundation for Representation, Inference, and Learning

**Song-Chun Zhu**
**University of California Los Angeles**

**JULY 2016**
**Final Report**



**STINFO COPY**

**AIR FORCE RESEARCH LABORATORY**
**SENSORS DIRECTORATE**
**WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7320**
**AIR FORCE MATERIEL COMMAND**
**UNITED STATES AIR FORCE**

# NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nationals.

AFRL-RY-WP-TR-2016-0127 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

// Signature//

Vince Velten, Program Manager
Electro-Optic Exploitation Branch
Layered Sensing Exploitation Division

// Signature//

Clare Mikula, Branch Chief
Electro-Optic Exploitation Branch
Layered Sensing Exploitation Division

// Signature//

Doug Hager, Deputy
Layered Sensing Exploitation Division
Sensors Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

*Disseminated copies will show "//Signature//" stamped or typed above the signature blocks.

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*



| 1. REPORT DATE *(DD-MM-YY)* | 2. REPORT TYPE | 3. DATES COVERED *(From - To)* |
|---|---|---|
| July 2016 | Final | 26 September 2011 – 15 March 2015 |

**4. TITLE AND SUBTITLE**
MATHEMATICS OF SENSING, EXPLOITATION, AND EXECUTION (MSEE)
Sensing, Exploitation, and Execution (SEE) on a Foundation for Representation, Inference, and Learning

**5a. CONTRACT NUMBER**
FA8650-11-1-7149

**5b. GRANT NUMBER**

**5c. PROGRAM ELEMENT NUMBER**
61101E

**6. AUTHOR(S)**
Song-Chun Zhu

**5d. PROJECT NUMBER**
1000

**5e. TASK NUMBER**
N/A

**5f. WORK UNIT NUMBER**
Y0P5

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
University of California Los Angeles
11000 Kinross Avenue, Suite 102
Los Angeles, CA 90095-0001

**8. PERFORMING ORGANIZATION REPORT NUMBER**

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
Air Force Research Laboratory
Sensors Directorate
Wright-Patterson Air Force Base, OH 45433-7320
Air Force Materiel Command
United States Air Force

Defense Advanced Research Projects Agency/DARPA/DSO
675 N Randolph Street
Arlington, VA 22203

**10. SPONSORING/MONITORING AGENCY ACRONYM(S)**
AFRL/RYAT

**11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)**
AFRL-RY-WP-TR-2016-0127

**12. DISTRIBUTION/AVAILABILITY STATEMENT**
Approved for public release; distribution unlimited.

**13. SUPPLEMENTARY NOTES**
This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to general public including foreign nationals. This material is based on research sponsored by Air Force Research laboratory (AFRL) and the Defense Advanced Research Agency (DARPA) under agreement number FA8650-11-1-7149. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation herein. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies of endorsements, either expressed or implied, of Air Force Research Laboratory (AFRL) and the Defense Advanced Research Agency (DARPA) or the U.S. Government. Report contains color.

**14. ABSTRACT**
This project developed a mathematical foundation for unified representation, inference, and learning for ISR problems. The result of the project is an end-to-end system for scene and event understanding from various imaging sensor inputs. It computes 3D scene-based probabilistic spatial-temporal-causal and-oor graph representations for human activities and human-object interactions, and answers binary (yes/no) queries via “Who, What, Where, When, and How” storylines.

**15. SUBJECT TERMS**
scene understanding, spatial-temporal-causal and-or graph, restricted visual turing test, computer vision, semantic description, 3D scene reconstruction, video synchronization, multi-view tracking, action recognition, reasoning with uncertainty

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT: | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON (Monitor) |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | Vincent Velten |
| Unclassified | Unclassified | Unclassified | SAR | 82 | 19b. TELEPHONE NUMBER *(Include Area Code)* N/A |

# Table of Contents

# 1. Executive Summary

In this section, we give an executive summary in terms of the testing data, the ontology and storyline-based queries, and the objectives, approaches and advantages developed by the team.

**Summary**:

This DARPA MSEE project developed a mathematical foundation for unified representation, inference and learning for ISR problems. The result of the project is an end-to-end system for scene and event understanding from various inputs. It computes 3D scene-based *probabilistic spatial-temporal-causal representations* for human and object activities, and answers queries via "Who, What, Where, When, and How" storylines. The system was 3rd party evaluated and tested using a Visual Turing Test (VTT) of *1000+ queries* on *100 hours* of recorded videos. In the on-going SIMPLEX project, this system is extended to support robot learning where an agent learns the Spatial, Temporal and Causal And-Or Graph (STC-AOG) from human demonstrations and refines the learned representation through situated dialogues. We elaborate details of the MSEE evaluation protocol in Section 2.1 and the system under test (SUT) developed by the UCLA grantee in Section 3 and Section 4.


Figure 1. Some snapshots of the MSEE testing data.

**Testing Data**: A 3rd party company, SIG, collected the multi-modality and multi-scene testing data under the DARPA contract. The data collection protocol was reviewed and approved by DARPA and Air Force Research Laboratory (AFRL). Figure 1 shows some snapshots of the testing data. The data are collected for multiple scenes including office, kitchen, lobby, meeting room, auditorium, parking lot area, and garden area across 4 seasons. More than 30 cameras are used and the length of recorded videos is about 100 hours. The number and placement of cameras were chosen carefully to capture the events and activities from a wide range of angles and distances including side views and bird's-eye views (cameras mounted on top of buildings). The views of cameras have moderate overlapping similar to typical surveillance settings. Moving cameras (hand-hold or mounted on cars/bicycles) are used to capture close-look details of objects, actions and events. We elaborate details of the MSEE benchmark in Section 2.2.

**Ontology and Storyline-based Queries**: We are interested in a selected ontology as listed in

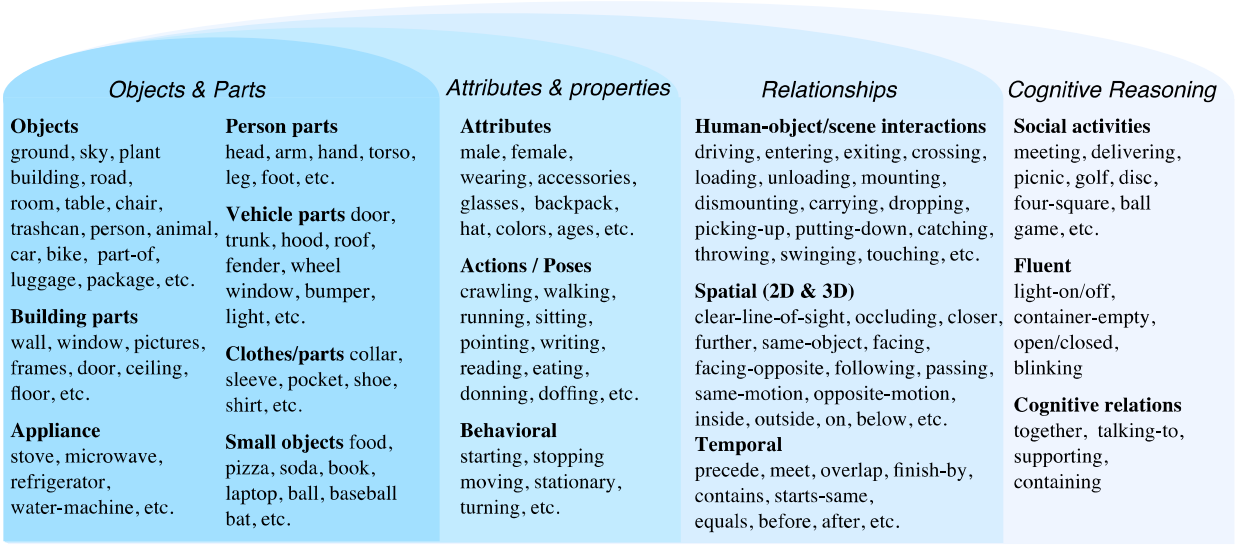



Figure 2. The ontology is sufficiently expressive to represent different aspects of spatial, temporal, and causal understanding in videos from basic level (e.g., identifying objects and parts) to fine-grained level (e.g., does person A have a clear-line-of-sight to person B?). Based on the ontology, we build a toolkit for collecting storyline-based queries and grounding annotations for each predicates. Queries organized in multiple storylines are designed to evaluate a computer vision system from basic object detection queries to more complex relationship queries, and further probe the system's ability in reasoning from the physical and social perspectives, which entails human-like commonsense reasoning. Cross-camera referencing queries require the ability to integrate visual signals from multiple overlapping sensors. Queries are stored in XML format. We elaborate details of the MSEE benchmark in Section 2.3.

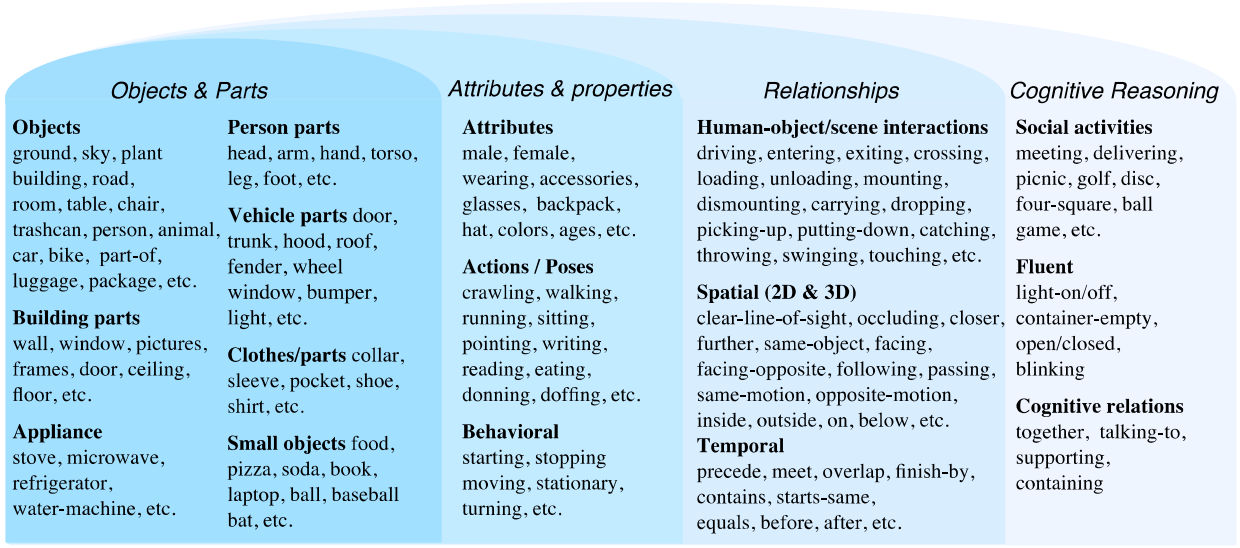



Figure 2. The table of MSEE ontology for deep understanding of scene and events in videos.

**MSEE Objective, Approach and Advantages**: Understanding video content consists of recognizing visible visual patterns and inferring hidden information (so called “dark matter”), where the former has been addressed well in the computer vision and machine learning literature

and latter still suffers from the lack of mathematically sound models and algorithms, as illustrated in Figure 3. The objective is to develop a unified and mathematically sound foundation for closing the gap in Sensing, Exploitation and Execution (SEE).

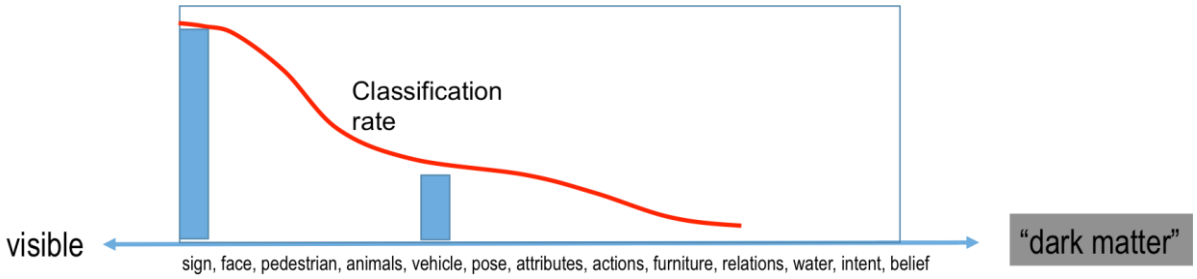



Figure 3. Illustration of the status of video understanding.

We developed a unified representation, the Spatial-Temporal-Causal And-Or Graph (STC-AOG). The STC-AOG is capable of,

- Supporting joint inference for all tasks in the ontology;
- Providing mutual context across the Spatial-Temporal-Causal dimensions;
- Learning from small training data in weakly supervised ways (e.g. Q/A); and
- Defining sensory concepts in mathematical terms (e.g., what is a chair, hammer, human, white old male, action, event?)

MSEE focused on spatial-temporal-causal reasoning with semantic hierarchical representation. The algorithms and systems are flexible and generalizable, which are designed as white boxes rather than black boxes so that the evaluation results can be diagnosed explicitly (as we presented the analyses of the results of Phase I, II and II evaluation in previous reports). Compared with popular methods in computer vision, our approaches developed during the MSEE project have the advantages as-follows:

| | MSEE | Popular Methods in Computer Vision |
|---|---|---|
| Evaluation | 3D scene-based relations | 2D image-based detection |
| | Storyline based queries | Classification and bounding boxes |
| Representation | Probabilistic, compositional, generative and interpretable | Convolutional features, discriminative and implicit |
| Learning | Weakly supervised training with small datasets | Supervised training with large-scale datasets |
| Inference | Bottom-up/top-down reasoning | Feed forward |

## 2. Evaluation Protocol and Benchmark Comparisons

In this section, we introduce the MSEE evaluation protocol – a restricted visual Turing test (VTT), and the MSEE benchmark. Briefly comparing with other popular datasets in the computer vision and machine learning community, we also show the challenges in the benchmark.

### 2.1. Evaluation Protocol – a Restricted Visual Turing Test

During the past decades, we have seen tremendous progress in individual vision modules such as image classification and object detection, especially after competitions like PASCAL VOC and ImageNet ILSVRC and the convolutional neural networks trained on the ImageNet dataset were proposed. Those tasks are evaluated based on either classification or detection accuracy, focusing on a coarse level understanding of data. In the area of natural language and text processing, there have been well-studied text-based question answering (QA). For example, a chatterbot named *Eugene Goostman* (https://en.wikipedia.org/wiki/Eugene_Goostman) was reported as the first computer program which has passed the famed Turing test in an event organized at the University of Reading. The success of text-based QA and the recent achievements of individual vision modules have inspired visual Turing tests (VTT) where image-based questions (so-called visual question answering, VQA) or storyline-based queries are used to test a computer vision system. VTT has been suggested as a more suitable evaluation framework in going beyond measuring the accuracy of labels and bounding boxes. Most existing work on VTT focus on images and emphasize free-form and open-ended QA's.

In the protocol, we are interested in a restricted visual Turing test setting with storyline-based visual query answering in long-term videos. Our scene and event understanding benchmark emphasizes a joint spatial, temporal, and causal understanding of scenes and events, which are largely unexplored in computer vision. By “restricted”, we mean the queries are designed based on a selected ontology (see

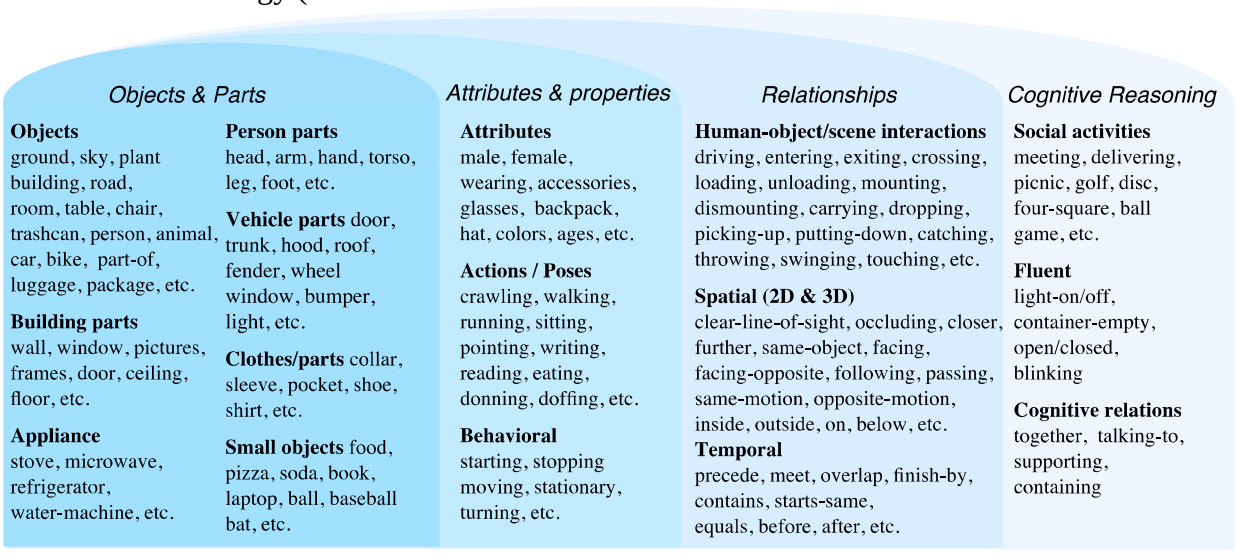



Figure 2).


Figure 4. Illustration of depth and complexity of the evaluation benchmark in scene and event understanding, which focuses on a largely unexplored task in computer vision -- joint spatial, temporal, and causal understanding of scene and event in multi-camera videos over relatively long time durations.

Figure 4 shows two examples in the benchmark. Consider the question how we shall test whether a computer vision system understands, for example, a conference room. In our benchmark, to understand a conference room, the input consists of multi-camera captured videos and storyline-based queries covering basic questions (e.g., $Q_1$, for a coarse level understanding) and difficult ones (e.g., $Q_k$) involving spatial, temporal, and causal inference for a deeper understanding. More specifically, to answer $Q_k$ in the office scene correctly, a computer vision system would need to build a scene-centered representation for the conference room, to detect, track, re-identify, and parse people coming into the room across cameras, and to understand the concept of sitting in a chair (i.e., the pose of a person and scene-centered spatial relation between a person and a chair), etc. The motivation is in three folds as follows.

- *Web-scale images vs. long-term videos.* Web-scale images emphasize the breadth that a computer vision system can learn and handle in different applications. These images are often of album photo styles collected from different image search engines such as Flickr, Google, Bing, and Facebook. This paper focuses on long-term, especially multi-camera captured, videos usually produced by video surveillance, which are also important data sources in the visual big data epic and have important security or law enforcement

applications. Furthermore, mutli-camera videos can facilitate a much deeper understanding of scenes and events. The two types of datasets are complementary, but the latter has not been explored in a QA setting.

- *Free-form and open-ended questions vs. restricted storyline-based queries*. In VQA, the input is an image and a ``bag-of-questions'' (e.g., is this a conference room?) and the task is to provide a natural language answer (either in a multiple-choice manner or with free-form responses). Free-form and open-ended questions are usually collected through crowd-sourcing platforms like Amazon Mechanical Turk (MTurk) to achieve diversity. However, it is hard to obtain well-posed pairs from a massive amount of untrained workers on the Internet. This is challenging even for simple tasks like image labeling as investigated in the ImageNet dataset and the Label-Me dataset. Currently, for the queries provided in the three-phase evaluation of MSEE, SIG adopts a selected yet sufficiently expressive ontology in generating queries. Following the statistical principles stated in the Turing test framework by Geman et al., we design an easy-to-use toolkit by which several people with certain expertise can create a large number of storylines covering different interesting and important spatial, temporal, and causal aspects in videos with the quality of queries and answers controlled. We are working on a more sophisticated toolkit and inspection methods to exploit MTurk to scale up collecting storyline-based queries covering long-term temporal ranges and across multi-cameras.

- *Quest for an integrated vision system.* Several methods proposed for image captioning and VQA are based on the combination of convolutional neural network and recurrent neural network like long short-term memory. In contrast to end-to-end approaches, in the MSEE system, we take an explicit approach to build a prototype system which integrates different vision modules, a knowledge base that manages visual parsing results, and a query engine that answers queries. The architecture supports symbolic reasoning on results generated by individual modules. We are interested in whether a computer vision system can further unfold the intermediate representation to explicitly show how it derives the answer, and if so it enhances the ``trust'' that we have on the system that it has gain a correct understanding of the scene.

## 2.2. Benchmark Challenges and Comparisons

### 2.2.1 Benchmark

In the benchmark, we organize data by multiple independent scenes (see Figure 1). Each scene consists of video footage from eight to twelve cameras with overlapping fields of view during the same time period. We have a total number of 14 collections covering both indoor and outdoor scenarios. Table 1 gives a summary of the dataset collected by SIG.

MSEE dataset reflects real-world video surveillance data and poses unique challenges to modern computer vision algorithms. We briefly summarize some typical challenges as follows. Figure 5 shows some snapshots (video clips are shown in the power-point report).

- **Varied number of entities**. In the dataset, activities in the scene could involve individuals as well as multiple interacting entities (see Figure 4).

| | Type | Cameras (Moving) | Length hh:mm:ss | Major events and activities |
|---|---|---|---|---|
| 1 | Indoor | 9 | 8:27:23 | Meetings, package exchange |
| 2 | Indoor | 12 | 17:35:36 | Meetings, card game, group lunch, coffee break |
| 3 | Indoor | 10 (1) | 2:29:50 | Classroom routines, lectures |
| 4 | Indoor | 11 (1) | 8:53:24 | Registration, classroom routines, lectures, evaluation |
| 5 | Outdoor | 9 (1) | 2:41:24 | Parking lot routines |
| 6 | Outdoor | 11 (2) | 8:15:44 | Parking lot routines |
| 7 | Outdoor | 9 | 2:22:00 | Four square game |
| 8 | Outdoor | 11 (2) | 8:14:42 | Various group ball games, bicycle races |
| 9 | Outdoor | 11 (1) | 13:15:06 | Various group ball games, auto repair |
| 10 | Outdoor | 11 (1) | 4:27:44 | Parking lot routines, auto repair |
| 11 | Outdoor | 7 (1) | 1:57:01 | Picnic, gardening, walking dogs |
| 12 | Outdoor | 10 (2) | 6:54:38 | Picnic, gardening, preaching |
| 13 | Outdoor | 8 (1) | 3:27:00 | Single-person exercises, ball and Frisbee games |
| 14 | Outdoor | 8 (2) | 4:15:56 | Group exercises, fashion contest, ball and Frisbee games |
| | | Total | 93.5 hours | |

**Table 1. Summary of the Benchmark.**

(a) Heavy Occlusion

(b) Pose Variations

(c) Illumination Variations

(d) Low Resolution and Heavy Shadow

Figure 5. Illustration of some typical challenges in the dataset (see video clips in the power-point report).

- **Rich events and activities**. The activities captured in the dataset involve different degrees of complexities: from the simplest single-person actions to the group sport activities which involve as many as dozens of people.

- **Unknown action boundary**. Unlike existing action or activity dataset where each action data point is well segmented and each segment only contains one single action, our dataset consists of multiple video streams. Actions and activities are not pre-segmented and multiple actions may happen at the same time. Such characteristic preserves more information about the spatial context of one action and correlation between multiple actions.

- **Multiple overlapping cameras**. This requires the system to perform multi-object tracking across multiple cameras with re-identification and 3D geometry reasoning.

- **Varied scales and viewpoints**. Most of our data are collected in 1920x1080 resolution, however, because of the difference in cameras' mounting points, a person who only occupies a couple of hundred pixels in bird's-eye views may occlude the entire view frame when he or she stands very close to a ground camera.

- **Illumination variation**. Areas covered by different cameras have different illumination conditions: some areas are covered by dark shadows whereas some other areas have heavy reflection.

- **Infrared cameras and moving cameras**. Apart from regular RGB cameras, MSEE dataset includes infrared cameras in some scenes as a supplementary (see Figure 1).

- **Moving cameras** (i.e., cameras mounted on moving objects) also provide additional challenges to the dataset and reveal more spatial structure of the scene.

### 2.2.2 Comparisons with Popular Datasets in the Computer Vision Literature

Conventional datasets are collected for testing individual task. while videos are comprehensive, and entail all modules to work together autonomously. If one module fails, it affects other modules.

(a)

(b)

**Figure 6. Some examples of person category in (a) PASCAL VOC and (b) ImageNet.**

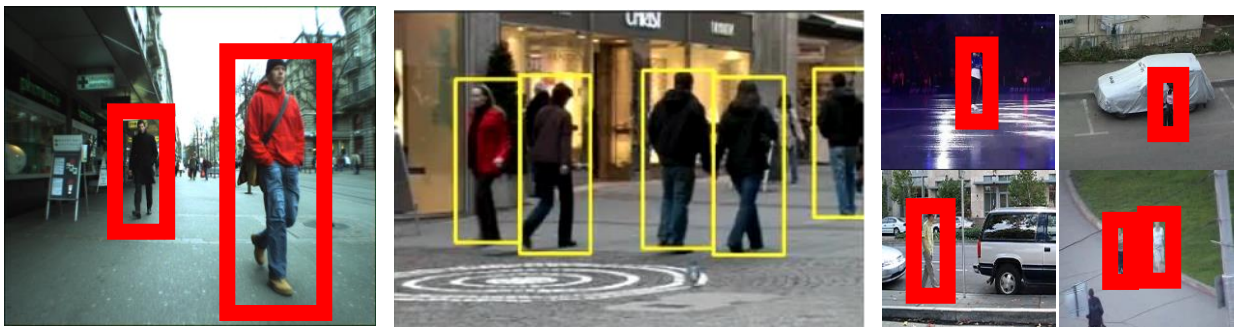
We briefly compare with some conventional vision benchmarks (including detection, tracking, re-identification, attributes, action and behavior) in the following, showing challenge and advantage of the MSEE dataset.

- *Comparison I: Object Detection*. In conventional image classification and object detection benchmarks, images are collected from photo albums, which are usually selected by users before uploading to the Internet. They are often well centered with good image quality. Figure 6 shows some examples of person category in two of the most popular vision benchmark, PASCAL VOC and ImageNet.

- *Comparison II: Object Tracking.* In comparing with conventional tracking benchmarks, the MSEE dataset excels in terms of the number of object entities, time range, resolution, occlusion and pose, as summarized in Table 2. We consider three popular tracking datasets. In the ETHZ tracking dataset (Figure 7 (a)), only the closest two people are required to track. There are few occlusions, and the resolution is good. Note that though the whole video has two minutes, tracking these two people only lasts shorter than 5 seconds. In the TUD tracking dataset (Figure 7 (b)), the whole video sequence has only 8 people in the sequence. There are only few occlusions and only three people cross each other. The whole video has only 1 minute, so there is no need for long-term tracking. In the MSEE dataset, in order to answer queries, long-term tracking is required. In this dataset, foreground (yellow) bounding boxes are *given*. In the TB-100 dataset (Figure 7 (c)), although there are large scale, illumination, pose variations and heavy occlusion, the task only requires to track one object at a time (on which the online And-Or graph tracker developed by the UCLA grantee obtains state-of-the-art performance, and we will elaborate quantitative results in Section 3).

| | #Objects | Time Range (min.) | Resolution | Occlusion | Pose |
|---|---|---|---|---|---|
| MSEE | 5 ~ 20 | 5 ~ 45 | Large variations | Heavy | Large variations |
| CV Benchmarks | < 10 | < 5 | Good | Few | Medium |

**Table 2. Tracking comparison between the MSEE dataset and CV benchmarks.**



(a) The ETHZ Datase (b) The TUD dataset (c) The TB-100 dataset

**Figure 7. Illustration of three popular tracking datasets.**

- *Comparison III: Object Re-identification across Views.* In conventional benchmarks of person re-identification across views, typical examples are well centered, cameras are from same angle, as illustrated in the left of Figure 8 (a). The MSEE dataset (see Figure 8 (b)~(d)) excels in many aspects as listed in Table 3.

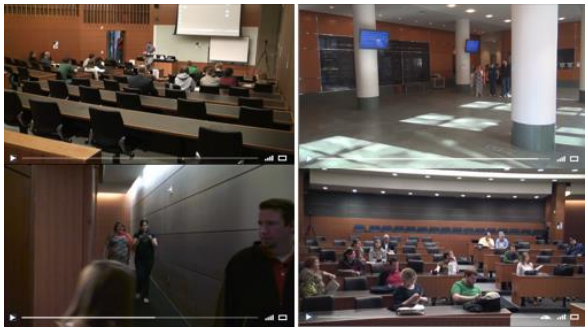
| | #Views | View Variation | Moving Camera | IR Camera | Pose | Working flow |
|---|---|---|---|---|---|---|
| MSEE | 4 ~ 8 | Both bird's-eye and person's-eye views | Yes | Yes | Large variation | 2D / 3D trajectory based |
| CV Benchmark | < 5 | All person's-eye view | No | No | Medium | Cropped image patch based |

**Table 3. Person re-identification comparisons between MSEE and other CV benchmarks.**

(a) The VIPeR dataset.

(b) MSEE Garden

(c) MSEE Parking-lot



(d) MSEE Auditorium

Figure 8. Left: The VIPeR dataset of person re-identification. It consists of 632 persons with 2 different view images per person. Right: three typical cases in MSEE. In each case, only four views are shown for clarity. To identify the same person across these 4 views are challenging, because the camera angles and distances are very different.

- *Comparison IV: Object Attribute.* Similarly, we summarize the comparisons in Table. 4. One of the most popular dataset is the Poselet dataset (see some examples in Figure 9) on which state-of-the-art performance is obtained by the UCLA grantee. We will elaborate quantitative results in Section 3.

**Figure 9. Some examples in the Poselet attribute dataset.**

| | **MSEE** | **Poselet Attribute Dataset** |
|---|---|---|
| # Attributes | 6 (Multi-class) | 9 (Binary) |
| Data | Video | Image |
| Resolution | Large variation | All high resolution |
| View | Large variation | Mostly frontal view |
| Bounding box | Given by detection | Ground-truth bounding box is given in testing |
| Occlusion | Heavy | Few |
| Pose | Large variations | Medium |
| Illumination | Large illumination and shadow | images are very clear |
| Scale | Large variations | Resized to similar size |

Table 4. Object attribute comparisons between MSEE benchmark and Poselet benchmark.

- *Comparison V: Action Recognition.* We summarize the comparisons between the MSEE benchmark and other CV benchmarks in Table 5. Figure 10 shows some examples in a popular CV benchmark, the Penn Action dataset.

| | #Actions | View variations | Moving camera | IR camera | Cross views | Working mode |
|---|---|---|---|---|---|---|
| MSEE | 4 ~ 20 | Both bird's-eye and person's-eye views | Included | Included | Yes | 2D / 3D trajectory based (segment issues due to variable – length of actions) |
| CV Benchmarks | 1 per video clip | Person's-eye or close-look views | Not | Not | No | Segmented video clips (still a classification problem) |

Table 5. Action recognition comparisons between MSEE benchmark and other CV benchmarks.

Figure 10. Some examples in the Penn action dataset.

- *Comparison VI: Behavior Recognition.* In conventional benchmarks, the task is to classify a video clip with one behavior label. In MSEE, we need to solve the temporal parsing problem in segmenting an input long video. We summarize the comparisons in Table. 6. Figure 11 shows some examples in three conventional benchmarks and the MSEE dataset.

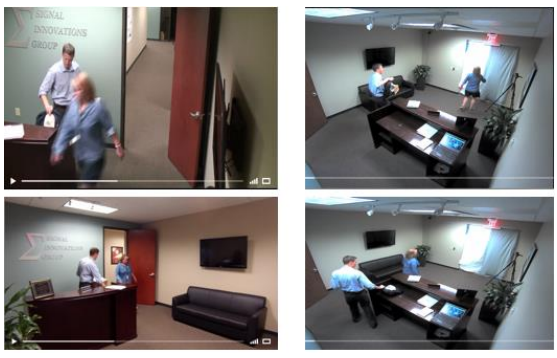
| | #Behavior | View variations | Moving camera | IR camera | Cross views | Working mode |
|---|---|---|---|---|---|---|
| MSEE | 4 ~ 20 | Both bird's-eye and person's-eye views | Included | Included | Yes | 2D / 3D trajectory based (segment issues due to variable –length of behavior) |
| CV Benchmarks | 1 per video clip | Either person's-eye or bird's-eye views | Not | Not | Not | Segmented video clips |

Table 6. Comparisons of behavior recognition in the MSEE benchmark and other CV benchmarks.


(a) Examples in the collective activity dataset


(b) Examples in the VIRAT dataset


(c) Examples in the UT-interaction dataset



(d) Examples in the office scene in MSEE (Passing, Following, Together). People are occluded by furniture, and behaviors are usually ambiguous, need high level cognitive reasoning.

Figure 11. Examples in different CV benchmarks and the benchmark.

### 2.2. 3. The Complexity and Challenge of Benchmark

To demonstrate the difficulties of MSEE benchmark, we conduct a set of experiments on a typical subset of data using the state-of-the-art deep learning based object detection models (i.e., the faster RCNN method) and multiple-object tracking methods. We summarize the results in Table. 7. We can see that state-of-the-art individual models perform poorly in the benchmark.

| Dataset | Fashion | Sport | Evacuation | Jeep |
|---|---|---|---|---|
| Cameras | 4 | 4 | 4 | 4 |
| Length (mm:ss) | 4:30 | 1:35 | 3:00 | 3:35 |
| Frames | 32,962 | 11,798 | 21,830 | 25,907 |

| Dataset | Fashion | | | | Sport | | | |
|---|---|---|---|---|---|---|---|---|
| Detection | 0.475 | 0.413 | 0.635 | 0.485 | 0.554 | 0.596 | 0.534 | 0.694 |
| Tracking MOTP | 0.683 | 0.674 | 0.692 | 0.694 | 0.728 | 0.727 | 0.716 | 0.739 |
| Tracking MOTA | 0.341 | 0.304 | 0.494 | 0.339 | 0.413 | 0.483 | 0.430 | 0.573 |
| | Evacuation | | | | Jeep | | | |
| Detection | 0.518 | 0.556 | 0.534 | 0.533 | 0.252 | 0.250 | 0.280 | 0.389 |
| Tracking MOTP | 0.698 | 0.692 | 0.720 | 0.651 | 0.680 | 0.651 | 0.689 | 0.696 |
| Tracking MOTA | 0.389 | -0.241 | 0.346 | 0.399 | 0.172 | 0.170 | 0.203 | 0.270 |

Table 7. Top: Summary of the selected subset of data in MSEE benchmark. Bottom: Results from detection and tracking. For Detection: AP of all object occurrence is calculated as in PASCAL VOC 2012 based on results by Faster R-CNN. For Tracking: MOTA and MOTP are calculated as in Multiple Object Tracking Benchmark.

### 2.3 Storyline based Queries

In this section, we first introduce the format of formal language queries and then present the collection of queries. To better describe the queries, we give a systematic overview of the MSEE system in Figure 12 with details of the joint parsing module to be presented in Section 3.

#### 2.3.1. Formal Language Queries

A formal language query is a first-order logic sentence (with modification) composed using variables, predicates (as shown by the ontology in Figure 2), logical operators ($\wedge$,$\vee$, $\neg$), arithmetic operators, and quantifiers ($\exists, \forall$). The answer to a query is either true or false meaning whether the fact stated by the sentence holds given the data and the system's state of belief. The formal language representation eliminates the need of natural language processing and allows us to focus computer vision problems on a constrained set of predicates.

We evaluate computer vision systems by asking a sequence of queries organized into multiple storylines. Each storyline explores a natural event across a period of time in a way similar to conversations between humans. At the beginning of a storyline, major objects of interest are defined first. The vision system under evaluation shall indicate whether it detects these objects. A correct detection establishes a mutual conversation context for consecutive queries, which

ensures the vision system and queries are referring to the same objects in later interactions. When the system fails to detect an object, consecutive queries regarding that object is skipped.

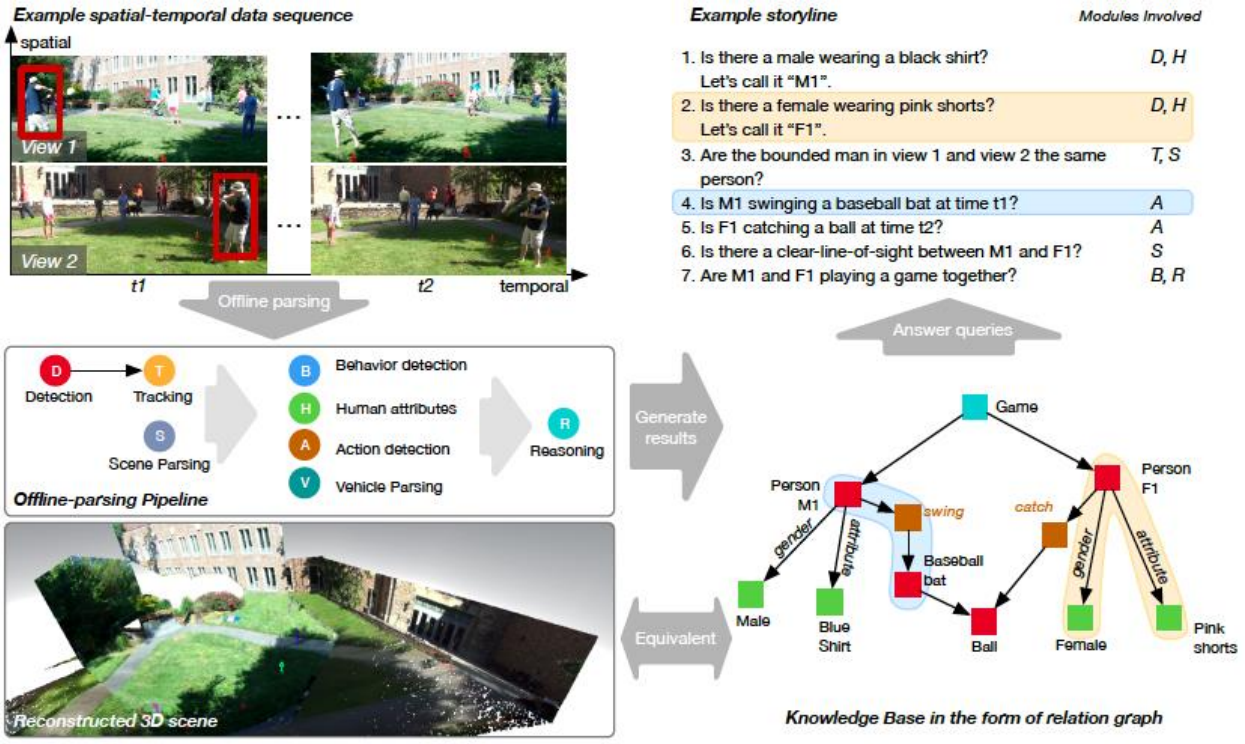



Figure 12. Illustration of the MSEE system under test (SUT). Top-left: input videos with people playing baseball games. Middle-Left: Illustration of the offline parsing pipeline which performs spatial-temporal parsing in the input videos, which we will elaborate in Section 3. Bottom-Left: Visualization of the parsed results. Bottom-Right: The knowledge base constructed based on the parsing results in the form of a relation graph. Top-Right: Example storyline and queries. Graph segments used for answering two of the queries are highlighted.

**Object predicates.** To define an object, specifications of object type, time, and location are three components. Object type is specified by object predicates in the ontology. A time *t* is either a view-centric frame number in a particular video or a scene-centric wall clock time. A location is either a point (x, y) or a bounding box (left-top corner point, $x_1$, $y_1$, and right-bottom corner point, $x_2$, $y_2$) represented by its two diagonal points, where a point can be specified either in view-centric coordinates (i.e. pixels) or in scene-centric coordinates (i.e. latitude-longitude, or coordinates in a customized reference coordinate system, if defined). For example, an object definition query regarding a person in the form of first-order logic sentence would look like:

$$\exists p \;\; person\left(p;\; time = t;\; location = (x_1, y_1, x_2, y_2)\right)$$

when the designated location is a bounding box.

**Attribute and relationship predicates**. Attribute and relationship predicates are used to explores a system's spatial, temporal, and causal understanding of events in a scene regarding the detected objects. The query space consists of all possible combinations of predicates in the ontology with the detected objects (and/or objects interacting with the detected ones) being the

arguments. When expressing complex activities or relationships, multiple predicates are typically conjuncted to form a query. For example, suppose $M_1$ and $F_1$ are two detected people, the following query states "$M_1$ is a male, $F_1$ is a female, and there is a clear line of sight (CLOS) between them at time $t_1$":

$$male(M_1) \wedge \ female(F_1) \wedge CLOS(M_1, F_1;\ time = t_1).$$

Note that the location is not specified, because once $M_1$ and $F_1$ is identified and detected, we expect the vision system can track them over space and time.

Moreover, storylines unfold fine-grained knowledge about the event in the scene as it goes. In particular, given the detected objects and established context, querying about objects interacting with the detected ones becomes unambiguous. As in the example shown in Figure 12, even the ball is not specified by any object definition queries (and actually it is hard to detect the ball even if the position is given), once the two people interacting with the ball are identified, it becomes legitimate to ask if "the female catches a ball at time $t_2$":

$$\exists\ b\ ball(b) \wedge\ catching(F_1, b;\ time = t_2),$$

and if "the male and female are playing a ball game together over the period of $t_1$ to $t_2$":

$$game(M_1, F_1;\ time = (t_1, t_2)).$$

Times and locations are specified the same way as in object definition queries with an extension that a time period ($t_1$, $t_2$) can be specified by a starting time and an ending time.

Correctly answering such queries is non-trivial as it requires joint cognitive reasoning based on spatial, temporal, and casual information across multiple cameras over a time period.

### 2.3.2. Collection of Queries


Figure 13. An example of composing queries using our query collection toolkit (developed after phase III evaluation, to be elaborated in Section 4).

The queries in three-phase evaluation are collected by an independent company, SIG. We re-implement an annotation tool with crowd-sourcing capability. We briefly introduce the procedure here with more details to be presented in Section 4. When composing a query, we first define and annotate the objects of interests. The annotation tool allows annotators to draw bounding boxes and points to refer to specific objects and move the annotated boxes along the video timeline to generate a ground-truth track. Tracks from different views can also be associated with same identity for collection cross-view tracking ground-truth. After the objects

are annotated, we obtain a list of object predicates with groundings. Next step is to compose queries by concatenating an arbitrary number of attribute or relationship predicates. Each predicate is annotated with a binary label ``true'' or ``false'' indicating whether the objects involved in the predicate satisfy the relationship, this serves as the grounding of attributes and relationships of objects. To ensure the collected queries are meaningful, we constrain the possible choices for each argument of a predicate so that the allowed combinations always represent conceptually correct relationships that align with commonsense. This lowers the bar for educating annotators and make it possible to adopt this tool to crowdsourcing platforms like Amazon Mechanical Turk. Figure 13 illustrates an example of this process. For each query, we also collect a ground-truth answer and a sentence that is the natural language equivalent to the first-order form.

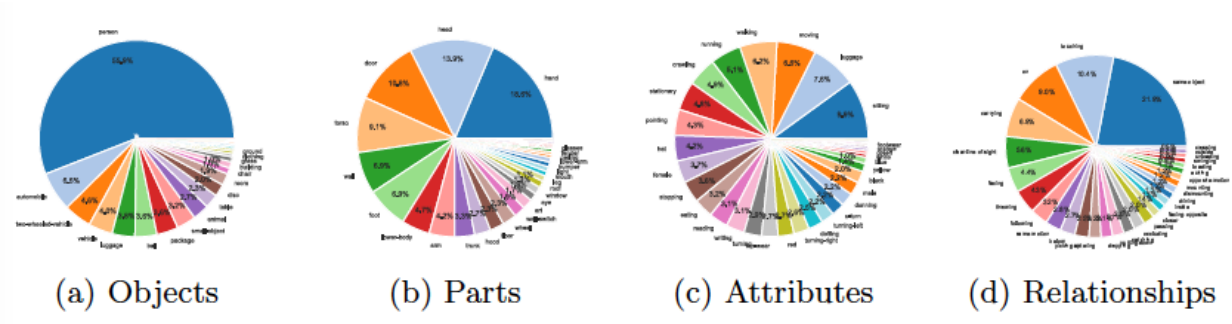



Figure 14. Distribution of predicates.

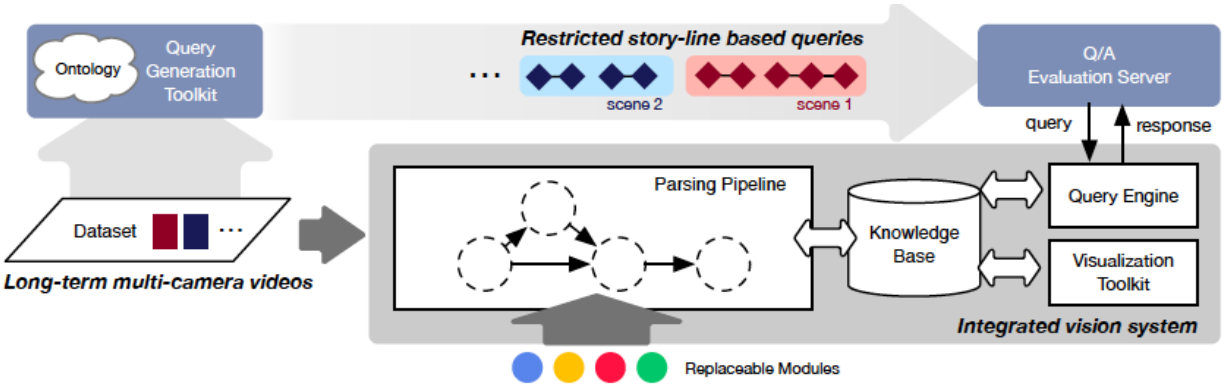
Currently, we have created 3,426 queries in the dataset. Figure 14 shows the distribution of predicates in selected categories. Though we try to be unbiased in general, we do consider some predicates are more common in and important than others and thus make the distribution non-uniform. For example, among all occurrence of object predicates, ``person'' takes 55.9%, which is reasonable because human activities are our major point of interest.

## 3. System for Test

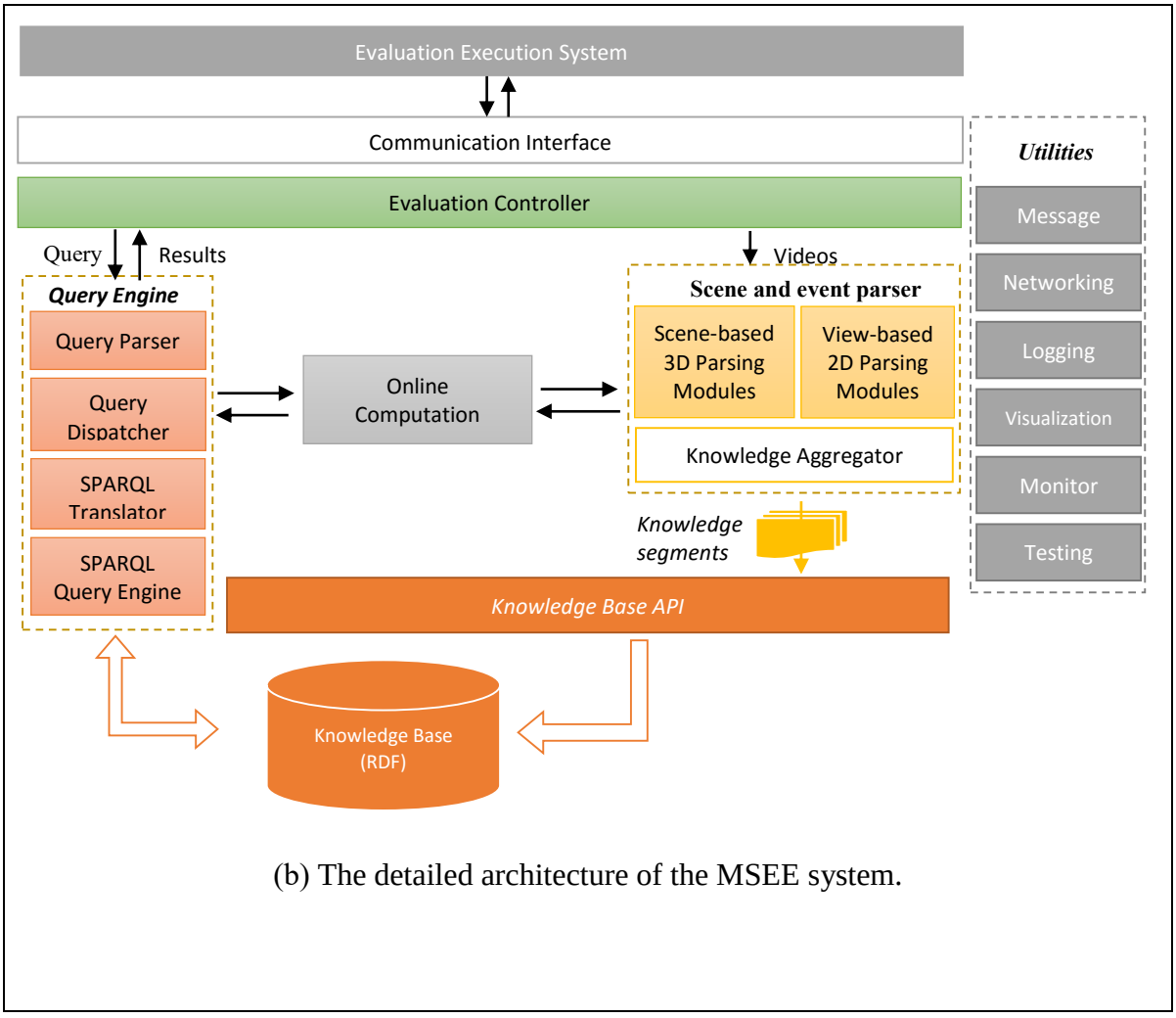
The system is built on the unified representation --- *spatial, temporal and causal and-or graph* (STC-AOG) developed in Phase I, and is tested extensively on public benchmarks and achieved state-of-the-art performances as we showed in publications. We have integrated these components into a system according to the Evaluation framework as well as the testing framework and videos provided by SIG.  This section summarizes the system.

### 3.1. System architecture and interface

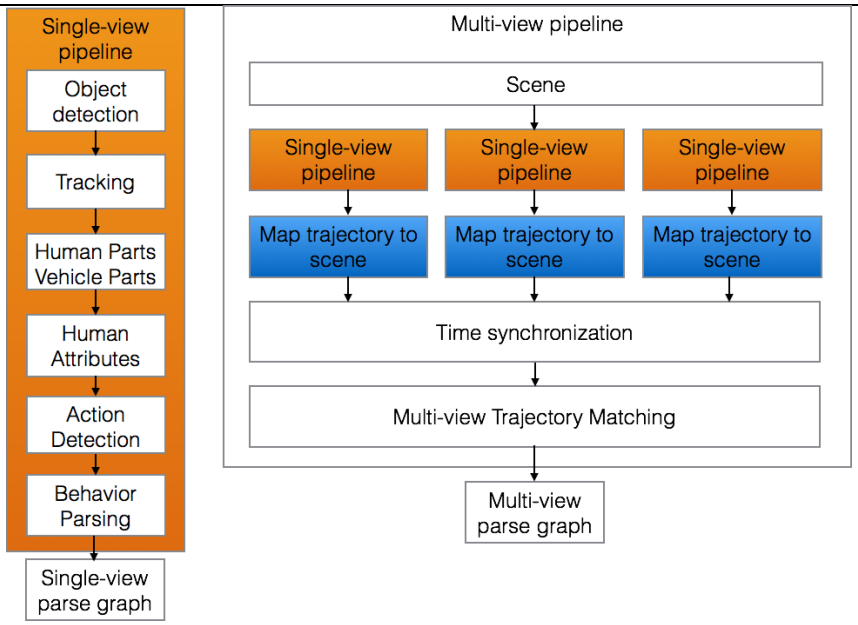
The system consists of three major parts: an offline parsing pipeline which decompose the visual perception into multiple sub-tasks, a knowledge base which stores parsing results (including entities, properties, and relations between them), and a query engine which answers queries by searching the knowledge base. The system also features a flexible architecture and a visualization toolkit.




(a) A systematic overview of the SUT developed by the UCLA team.




(b) The detailed architecture of the MSEE system.




(c) The vision pipeline system

Figure 15. The MSEE system.

The overall architecture of our system is shown in Figure 15. The system contains four modules.

I. **Evaluation controller** is the central component in our system which controls the overall workflow, coordinates all other components, and communicates with the evaluation execution system from SIG side. It ingest the video input from many cameras over multiple scenes.

II. **Scene and event parser** is the core component that parses the input videos and is composed of many vision components. All parsers are partitioned into two groups:

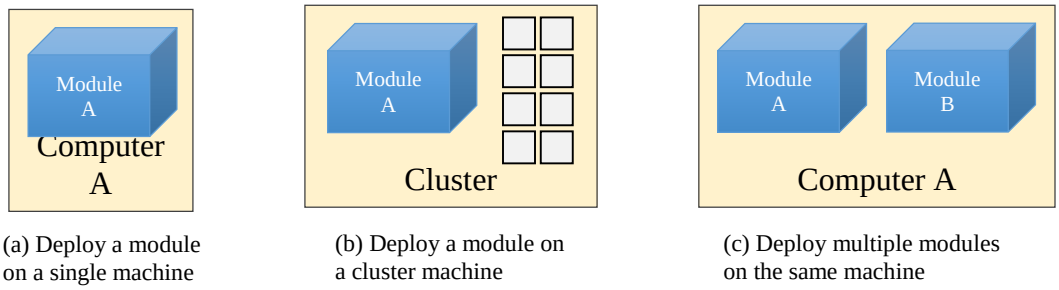
- *View-based 2D parsing modules*: parsing the scenes, objects, actions and events for each individual camera view.
- *Scene-based 3D parsing modules*: 3D scene reconstruction, and multi-view registration, and estimating the 3D coordinates in the world frame for objects in 2D image views.

Due to the difference in perspectives, 2D parsing information of one object computed from one view may conflict with the parsing obtained from another view. *Knowledge aggregator* aims to resolve such conflicts at the finally stage after all views are registered.

III. **Knowledge base and query engine** stories the entire pre-computed parse graphs about the scene in a RDF format and queries this knowledge base using SPARQL scripts. A *query translator* is used to translate formal language queries into appropriate SPARQL scripts.

IV. **On-line computing modules** are called by a query dispatcher and thus compute the remaining parts of the parse graph which are not computed prior to the query. For example, whether a person A has a clear-line-of-sight to see an object B, or whether person A and B are facing each other. Since it would be too computational intensive to pre-compute all possible relations, we compute them only when they are queried.

Our system is designed to be *distributed* and *cross-platform* and runs on a cluster machine with 48 CPU cores and a large number of GPU units. Since many of the parsing tasks are computationally expensive, each individual module can be deployed either on a single computer or on a cluster according to its computational cost and performance constraint. It is also possible to have multiple modules deployed on the same machine as long as they can be operating together efficiently. Figure 16 illustrates the choices of deployment. Such design gives us the flexibility to allocate resources.




(a) Deploy a module on a single machine

(b) Deploy a module on a cluster machine

(c) Deploy multiple modules on the same machine

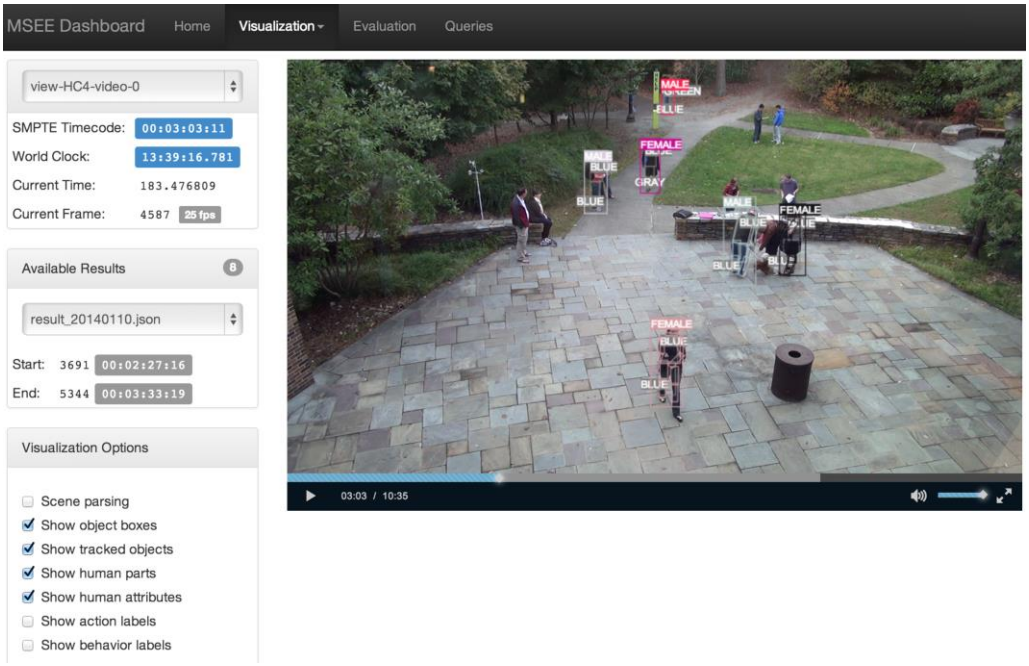
Figure 16. Three ways to deploy a module.

Offline parsing pipeline processes the multiple-view videos. Each view is first processed by a single-view parsing pipeline (Figure 15 (b)) where video sequences from multiple cameras are handled independently. Then multiple-view fusion matches tracks from multiple views, reconciles results from single-view parsing, and generates scene-based results for answering questions. To take advantage of achievements in various sub-areas in computer vision, we organize a pipeline of modules, each of which focuses on one particular group of predicates by generating corresponding labels for the input data. Every module gets access to the original video sequence and products from previous modules in the pipeline. The implemented modules are described as follows. Most components are derived from the state-of-the-art methods at the time we developed the system and are pre-trained on other datasets.

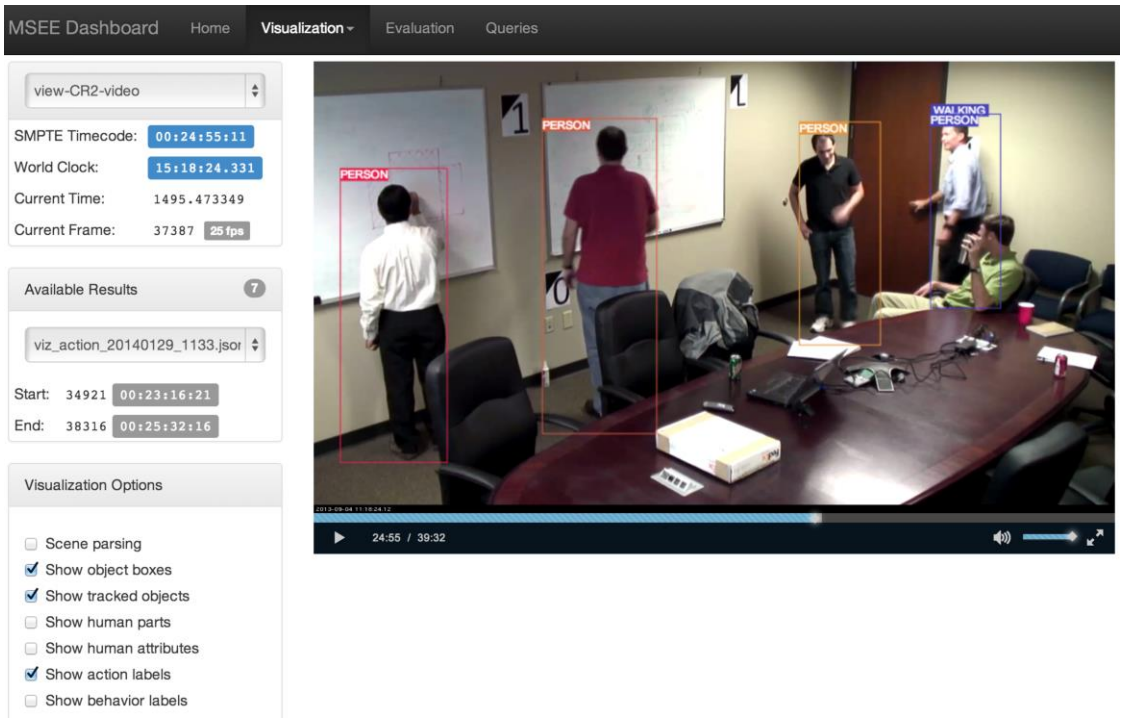
- **Scene parsing** generates a homography matrix for each sensor by camera calibration and also produces estimated depth map and segmentation label map for each camera view.
- **Object detection** processes the video frames and generates bounding boxes for major objects of interest.
- **Multiple object tracking** generates tracks for all detected objects.
- **Human attributes** classifies appearance attributes of detected human including gender, color of clothes, type of clothes, and accessories (e.g. hat, backpack, glasses).

- **Human Pose estimation** infers the locations of various parts of detected humans, including hands, arms, legs, foot, etc.
- **Action detection** detects human actions and poses in the scene.
- **Behavior detection** parses human-human, human-scene, and human-object interactions.
- **Vehicle parsing** produces bounding boxes and fluent labels for specific parts of detected cars (e.g. fender, hood, trunk, windows, and light).
- **Multiple-view fusion** merges the tracks and bounding boxes from multiple views based on appearance and geometry cues.

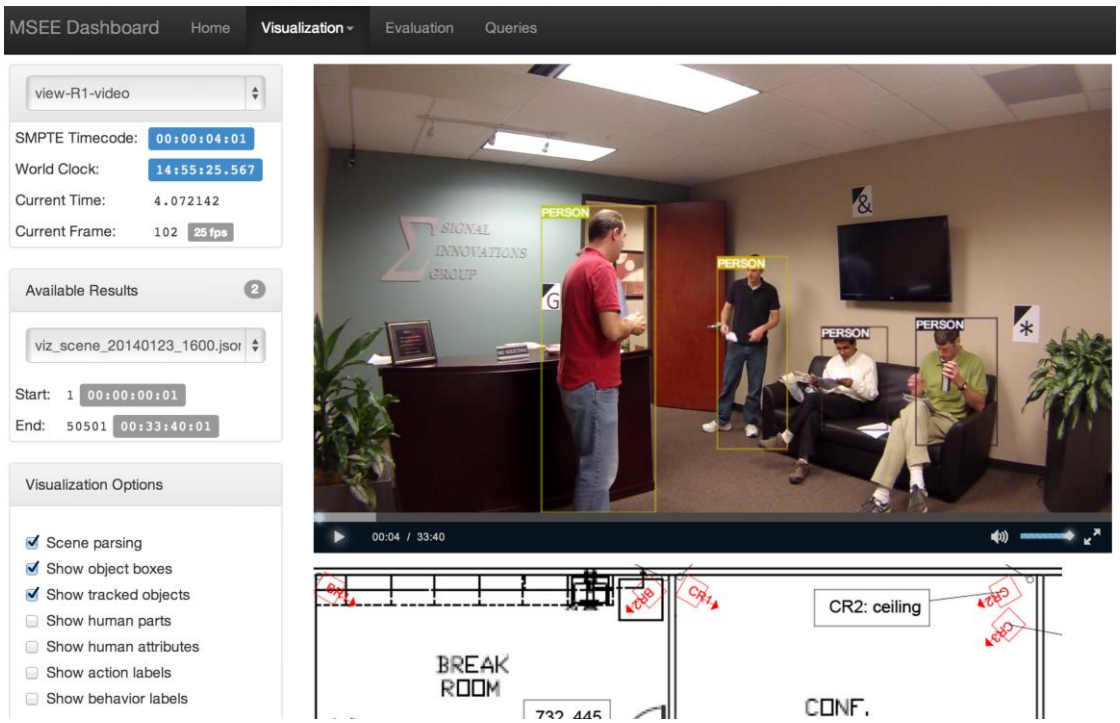
A dashboard has also been built to monitor system status and visualize parsing results. Figure 17 shows some screenshots on the SIG test scenes.




(a) Parsing the garden scene: we can test the system by selecting individual modules.




(b) The meeting room (the person sitting with green shirt is missed in this example).




(c) The lobby scene

Figure 17. screenshots of the dashboards for some scenes in the SIG test videos.

### 3.2. Preprocessing of videos

The videos from SIG are recorded by a network of 30+ cameras over three scene areas: a parking area, indoor with 5 rooms (hallway, lobby, meeting room, breakroom/kitchen, lecture hall), and a garden area. Before parsing the scenes and events, we compute three components in a preprocessing stage so that these videos are registered to a common space and time.

- *Camera calibration*: estimating the parameters for 4x4 projection matrices for each camera. These matrices transform between a pixel in the 2D image coordinates and a point in the 3D world coordinates.
- *Geo-registration*: estimating the parameters in a 3x3 homograph matrix by matching the ground seen by each camera to a common map so that we can track an object across multiple cameras.
- *Time-synchronization*: aligning the frames (individual images) of the videos from all cameras to the same time axis.

#### 3.2.1. Camera calibration

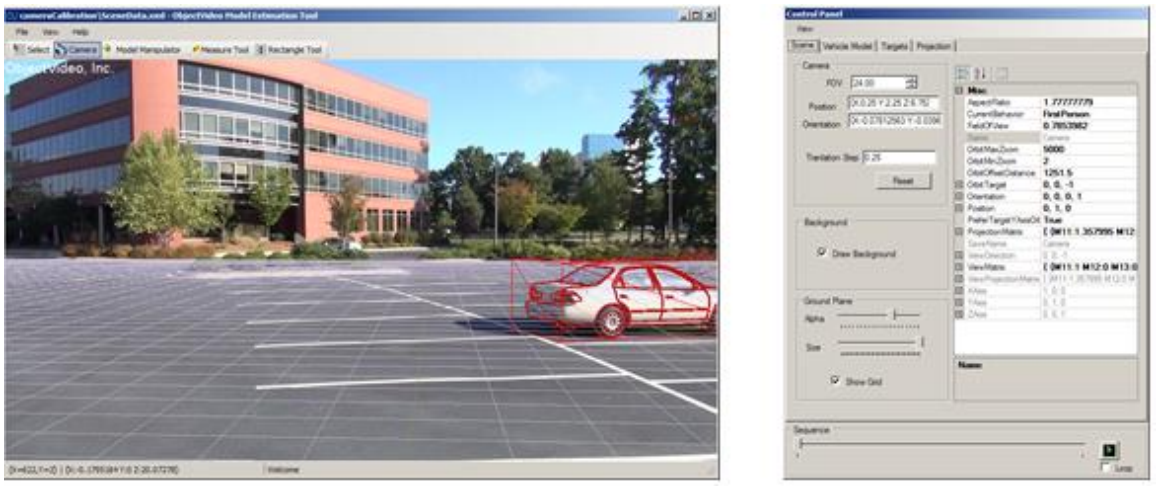


Figure 18. Camera calibration using known object models. (left) The fitted vehicle using known 3D CAD vehicle model; and (b) The camera calibration tool with estimated camera parameters.

For each camera, we estimate a *projection matrix.* This 4x4 matrix transforms a pixel in the 2D image frame to the 3D world frame. Traditionally this is done by putting a simple object of known shape, such as a checkerboard or a cube in the scene. By waving the checkerboard at the beginning of the video, one gets a number of corresponding points (minimum 8 points) to estimate the matrix. This is restricted when the checkerboard is not available. In our project, we develop a method to estimate the projection matrix by common objects in the scene, such as a human or a vehicle with a known height rather than using the calibration checkerboard. We collected accurate 3D CAD vehicle models over 260 different makers and models. Even though a specific vehicle in the scene might not be in the vehicle database, the fitting of a 3D CAD

model of a similar vehicle would still provide accurate estimates of the camera parameters. Figure 18 shows the camera calibration tool and the fitted 3D CAD model of a car in the SIG parking lot.

### 3.2.2. Geo-registration

For each camera, we also estimate a *homograph matrix*. This 3x3 matrix transforms a pixel (i.e. 2D image coordinate) from a camera view to a point (longitude, latitude) on the common ground plane. It is estimated by matching pixels on the ground in an image to the corresponding points in the common map. Currently we use the google map for the outdoor areas and using the floor plan for indoor areas. To find the corresponding (pixel, point) pairs, we can either match the pixels and points with distinct features. When this is not available (e.g. the ground image is flat without distinct features), an engineering method is to track a mobile object with GPS unit. The pixels of the detected feet of a walking person are matched to his/her GPS coordinate on the map. Figure 19 and 20 show the geo-registration of some indoor and outdoor images to the common ground maps. During phase II, we also implemented and improved the module for geo-registration of aerial videos. This is used for the UAV video and MAMI videos that we showed in the Q8 report.

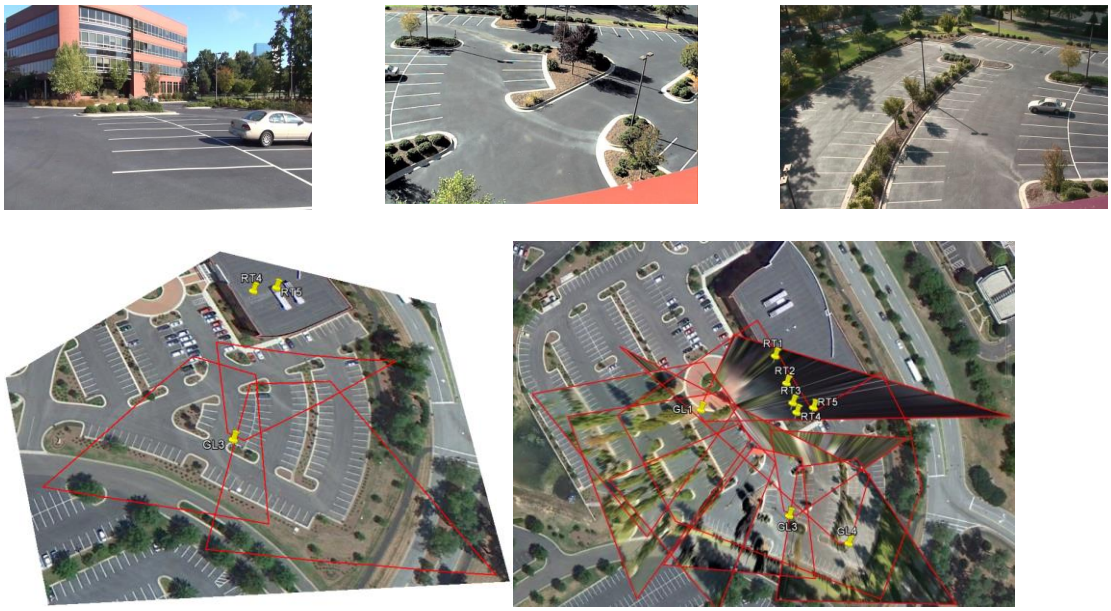



Figure 19. Geo-registration of three camera views (top) at the parking lot scene. (bottom-left) The three pins show the camera positions and the red polygons are the ground seen by the three cameras. (bottom-right) More views are registered to the map.

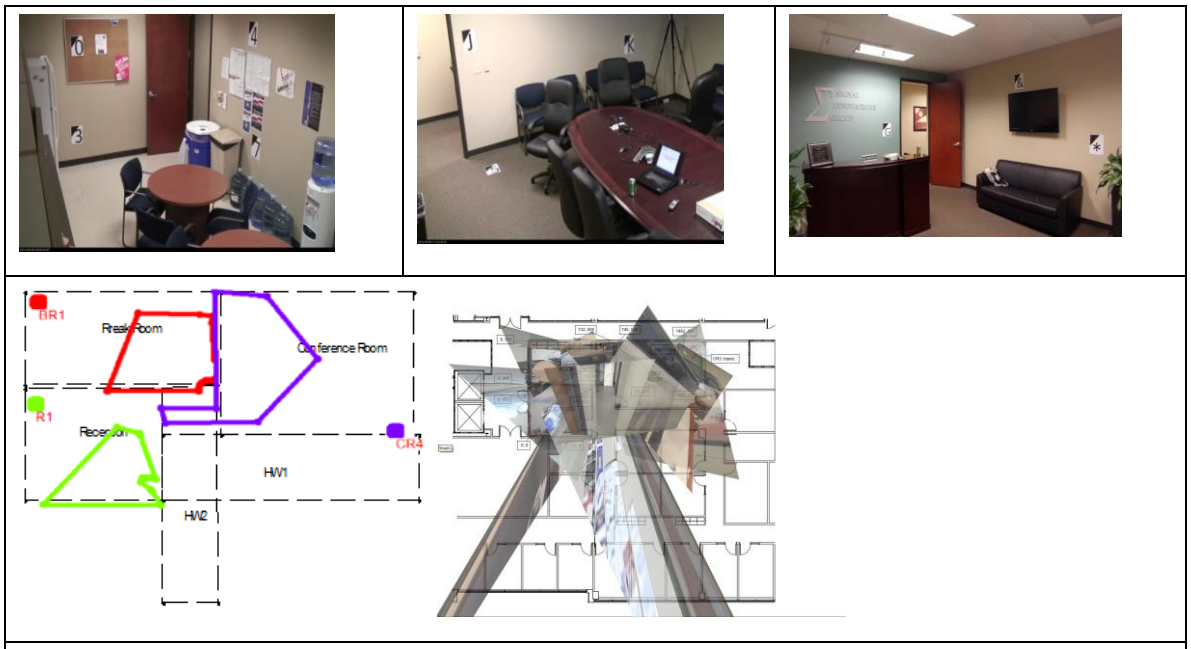



Figure 20. Geo-registration of three camera views inside the building (top). (bottom-left) The three colored dots show the camera positions and the polygons are the ground viewed at the three cameras. (bottom-right) More views are registered to the floor plane.

### 3.2.3. Time synchronization

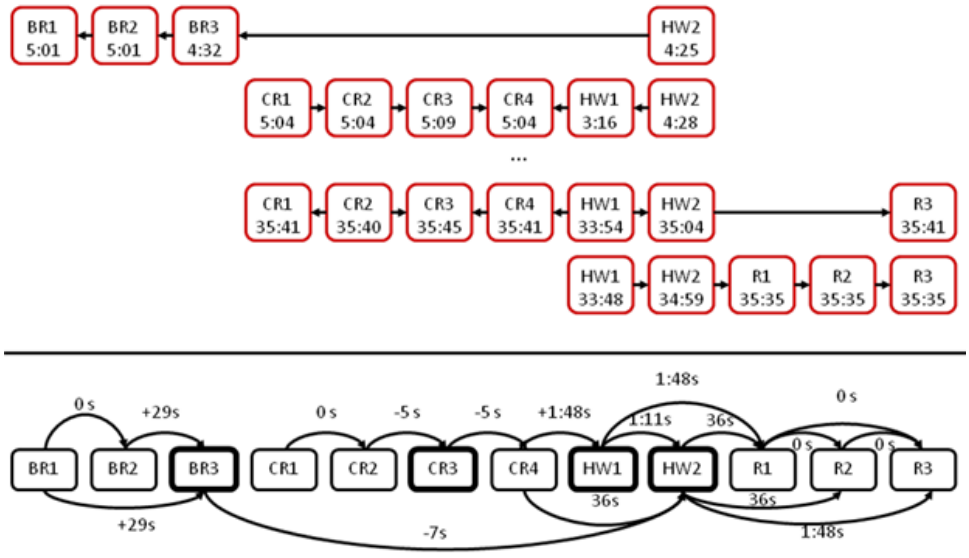



Figure 21. (Top panel) Time synchronization using objects appeared on multiple cameras (nodes) which have overlapping FOVs. Each row indicates the camera and the time when a target appeared in the same geo-location. (Bottom row) the synchronization results between 12 cameras. The value on the edge indicates the estimated time difference of two cameras with overlapping FOVs.

The videos provided by SIG are not time-synchronized. Since time synchronization is critical in multi-view target tracking, we developed an algorithm to estimate the time offsets and frame rates of videos from all cameras. From the camera calibration, the overlapping FOVs between two cameras are determined after geo-registration. When an object is detected and tracked, its footprint (a key point at the bottom of the bounding box) is placed on the common ground plane. When they are in the area seen by two cameras, such footprints provide the frames for synchronizing the two cameras.

Figure 21 illustrates the process. We use the tracking results of each camera, and capture the *appear* events on the overlapping area of two cameras. For example, in the first row of Figure 21, a person with a red shirt appears on camera HW2 at 4:25, on camera BR3 at 4:32, and on cameras BR2 and BR1 at 5:01. When the target is in the area overlap, the time should be the same. We continue to record *appear* events of the red shirted person as well as the time differences of the camera pairs. When tracking is over, we collect all time differences and find a time offset and a frame rate of each camera to minimize the sum of time differences. This value should be zero when the target is tracked perfectly across all cameras. The optimization is done by changing the time offset and frame rate of cameras until it converges. The estimated time differences between each pair of cameras pairs with overlapping FOV are shown in the last row of Figure 21.

By propagating the time differences among overlapping cameras in the SIG dataset, we can sync all cameras as there are overlapping FOVs among these camera pairs. However, when the cameras are not fully "connected" through the overlapping FOV relations, the time difference between *disappear* event of camera A and *appear* event of camera B need to be recorded over time. Then we can estimate the average time difference for time synchronization.

### 3.3. Spatial parsing

Spatial parsing includes a ranges of tasks in computing the scene-object-parts hierarchy:

- Parsing 3D scenes as the context for detecting object and recognizing actions;
- Parsing human figures – human detection, pose estimation and attribute recognition;
- Parsing vehicle – detection, pose/view estimation, and attributes recognition; and
- Detecting and recognizing other objects in the scene: animals, furniture etc.

#### 3.3.1. Parsing 3D scenes

A scene consists of several functional areas for human/vehicle activities, and thus parsing the scene into various parts (wall, floor, doors, table, chairs etc.) provides crucial contextual information for detecting and recognizing human, vehicle and other object, and for understanding their actions and events.

3.3.1.1 Constructing 3D Scene from a single view

Traditional method constructs 3D scenes (depth) from multiple views using perspective geometry constraints. However, human vision can perceive a 3D scene from a single 2D images.

Though this 3D construction may not be very accurate due to ambiguity, it is often sufficient to support the reasoning of functionality and actions.

We have implemented such a module by exploiting commonsense knowledge, for example, a) buildings are standingly upright; b) parallel lines in the world merge at vanishing points in images; c) man-made scenes (like a city block) usually observe the Manhattan structure where the lines in the scene form the X,Y,Z axes, and so on.

Figure 22. (a) Input image overlaid with detected parallel ines; (b) segmentation of scene layout; (c) synthesized image from a novel viewpoint; (d) recovered depth map (darker pixels means closer).

Figure 22 illustrates one typical result of our method. Given an input image in Fig.22(a), we detect line segments and cluster them in a few groups shown in color. Line segments in the same color are parallel in 3D space and point to the same vanishing point. In this example, there are 4 colors for 4 vanishing points. In general, three vanish points form a manhantten structure, i.e. a 3D world coordinate frame. So, by grouping these colored line segments into coherent 3D world frames, we can compute the 3D depth in Figure 22.(b). To illustrate the 3D depth, we generate the scene from a new views and the image is shown in Figure 22.(c).

The model underlying this method is an attributed and-or graph representation, and Figure 23 shows the parse graph for a scene. The And-or graph consists of a number of productions rules for how scene structures are decomposed in a hierarchical way. We augmented this graph by associating each node in the hierarchy some geometric attributes, which are the respectively.

- Vanishing points VP associated with the line segments;
- Cartisian coordinate system CCS associated with a local 3D world frame (Manhantten);
- Camera focal length associated with the root for the whole scene.

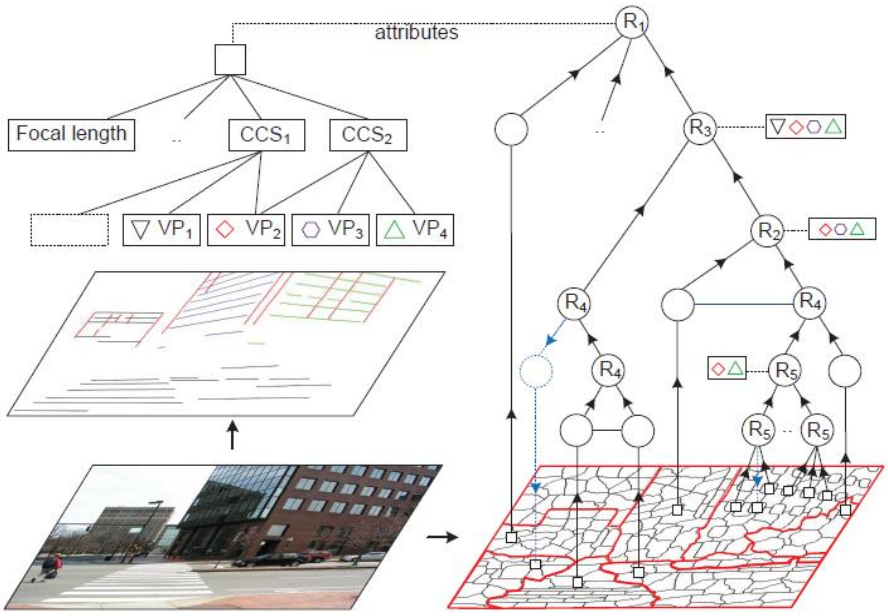



Figure 23. Scene parsing with attributed and-or graph (grammar). The right side is the parse graph derived from an And-or graph grammar, and the left are the augmented geometric attributes, which are associated with nodes in different levels of the parse graph.

We develop an effective top-down/bottom-up cluster sampling procedure to explore the constrained space efficiently and compute the hierarchical parse graph by recursively applying the grammar rules while preserving the attributes constraints. We evaluated our method on public benchmarks and achieved significant performance improvement over the existing methods. Figure 24 shows the parsing results on the two SIG scenes one outdoor and the other indoor.

Figure 24 . Results on the SIG scenes. (left) Input images; (right) Synthesized images of the computed 3D scenes from novel viewpoints to show the 3D effects.

The 3D scene reconstruction is integrated with the geo-registration to provide the scene context. Based on the geo-registered map, we can visualize the object trajectories on a common ground plane. Figure 25 illustrate three trajectories (one person walking and two people riding bikes) in the parking lot area.

Figure 25. Tracked object trajectories transformed from the 2D camera view to the 3D world coordinate frame.

3.3.1.2. Computing the Clear-line-of-sight and other 3D relations

The test framework also requests the computation of clear-line-of-sight which is to determine whether one person can see another person or object in the scene. This requires the full 3D models of all objects/surfaces in the scene. Other 3D relations, such as "On", "below", "Occluding" are also implemented based on the 3D reconstructed scene. We use very coarse 3D shapes, such as a 3D cuboid to represent objects in the scene, such as trees, table, chairs, because the precise 3D shape is infeasible to infer and time consuming. These 3D relations are computed online when they are asked by the query engine. The performance of this module still needs improvements.

### 3.3.2. Human figures, body parts and poses

Our work on and-or graph has achieved state-of-the-art performance on two big public dataset for human pose estimation: the UC Irvine PARSE dataset and the Leeds dataset. The results were published at CVPR June 2013. Table 1 below shows the accuracy of the detected body parts against human annotated results. Our results are compared against two other top performers in the literature. The main difficulty is with the lower arms and lower leg which are often occluded. Figure 26 shows some visual results of the pose estimation.

| Table 1 | Method | torso | head | upper leg | lower leg | upper arm | Lower arm | average |
|---|---|---|---|---|---|---|---|---|
| UCI PARSE | TZN (2012) | 97.1 | 92.2 | 85.1 | 76.1 | 71.0 | 45.1 | 74.4 |
| | YR (2011) | 97.6 | 93.2 | 83.9 | 75.1 | 72.0 | 48.3 | 74.9 |
| | Ours (AOG) | **99.5** | **97.4** | **89.2** | **78.3** | **74.6** | **56.9** | **79.5** |
| Leeds | TZN (2012) | 95.8 | 87.8 | 69.9 | 60.0 | 51.9 | 32.9 | 61.3 |
| | YR (2011) | 88.1 | 74.6 | 74.5 | 66.5 | 53.7 | 37.5 | 62.7 |
| | Ours (AOG) | **98.3** | **92.7** | **86.8** | **78.2** | **70.2** | **45.1** | **75.2** |


Figure 26 Typical detection results on the PARSE and Leeds dataset. Each red bounding boxes are the computer detected body parts. The last row shows some failure examples where the blue boxes are wrong detections. The arms and legs are still a challenge.

The MSEE task includes image sequences that present additional challenges for human parsing beyond typical pose estimation benchmarks. In particular, the method must handle prominent self-occlusion, and occlusion from external objects such as chairs, tables, and other people. Because a large portion of the videos are from indoor scenes. Below is a brief summary of our innovations for addressing these problems.

- **Self-occlusion**: In addition to the MSEE training data, we have annotated our own human pose and attribute dataset, described in the Q6 report, containing 2000 hi-res images annotated with part labels that include 2.1D depth. These depth annotations are used to determine if a part is hidden due to self-occlusion, and are used to train additional AND rules in the grammar. Correctly modeling the occlusion relationships between each pair of parts is computationally intractable, however, due to the cyclical dependencies that are created. Instead, the occlusion labels are made dependent only on the local configuration and appearances of the parts. This allows the grammar to learn better appearances templates and utilize local dependencies between both visible and occluded variants of each part.

- **External occlusion and multiple output configurations**: Occlusions from external objects often obscure large portions of the body, which cannot be captured well by the part-based occlusion productions used in the self-occlusion case. Instead, our and-or graph is redesigned to also derive partial configurations corresponding to typical occlusion modes, such as when only the upper body or head is visible. These occlusion modes appear under an OR in the grammar, and compete with each other to explain the image appearance.
- **Activity classification**: Static pose can provide strong evidence for certain activities, such as standing, sitting, crawling or lying down. The composition rules of the grammar can capture many of the local cues that are unique to these poses, such as the appearance of a part, its local geometry, and kinematic behavior between connected parts. By placing top-level rules in the grammar that are specific to these activities, the grammar can be trained to select the activity by combining evidence from these cues that come from lower-level compositions. This training is done under an appropriate loss function that penalizes the incorrect activity selection.

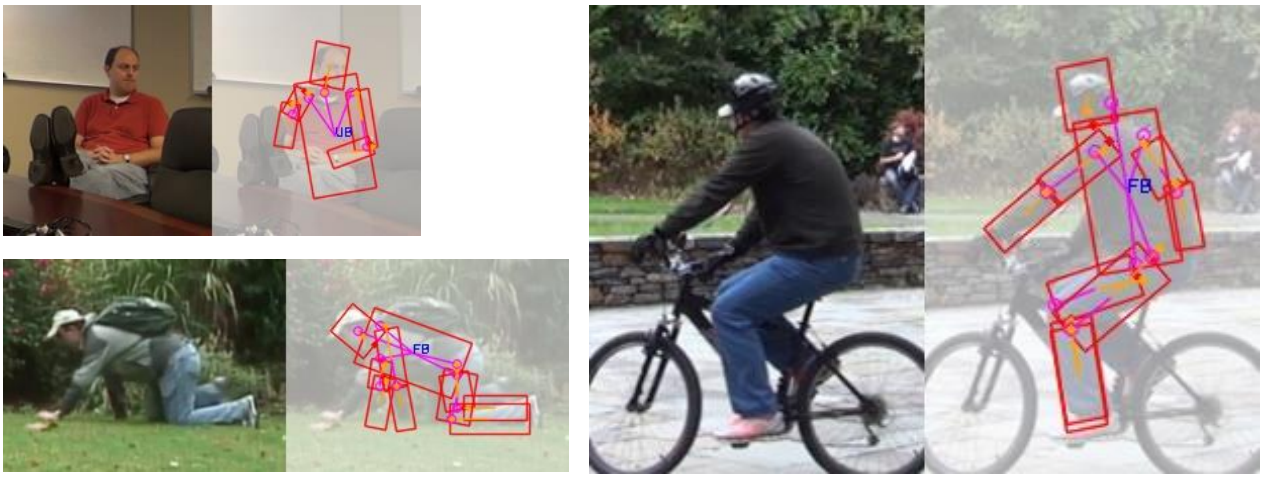



Figure 27. Some human pose estimation results on the SIG videos: sitting, crawling and riding a bike. The results are not perfect, especially for the arms and legs due to heavy occlusions. The poses can be also used to assist action recognition.

### 3.3.3. Human attributes

As we reported in Q6, we have collected 4,000 images of human and annotated corresponding pose and attributes. Some examples are shown in Figure 28. We have also developed a method for human attribute recognition, such as classifying a person as male or female, wearing a hat, glass, and the type of clothes and colors. Table 8 below is the performance tested on a public dataset against the poselet method by the UC Berkeley vision group. The report was published at CVPR 2013.

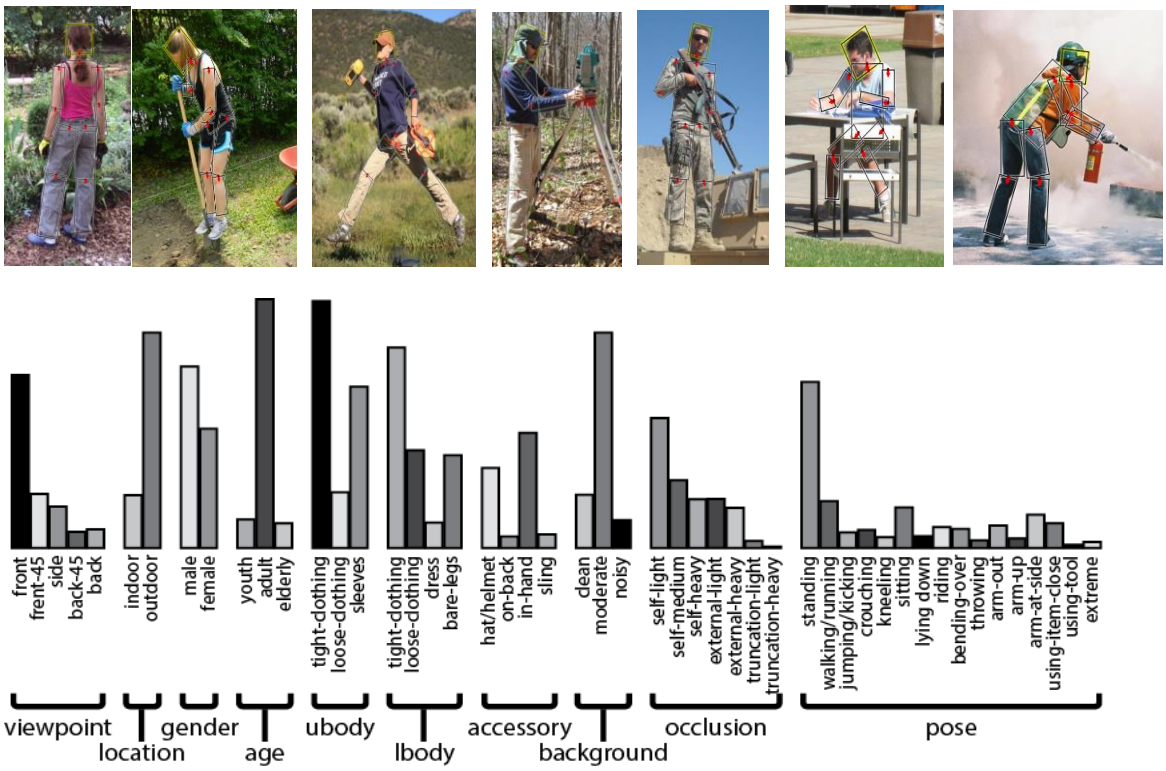



Figure 28. The dataset we collected includes 4,000 examples with human pose and attributes annotated. The body parts and attributes are labeled beyond the SIG evaluation.

| | Gender | Long Hair | Eyeglass | Hat | T-shirt | Long Sleeve | Shorts | Jean | Long Pant | Mean |
|---|---|---|---|---|---|---|---|---|---|---|
| Ours | 88.0 | 80.1 | 56.0 | 75.4 | 53.5 | 75.2 | 47.6 | 69.3 | 91.1 | 70.7 |
| Poselet | 82.4 | 72.5 | 55.6 | 60.1 | 51.2 | 74.2 | 45.5 | 54.7 | 90.3 | 65.2 |

Table 8. Human attribute recognition performance evaluation on a public benchmark.

We have extended our work in two ways to be presented in Section 4.

- **Developing a full attributed and-or graph for human attributes.** This model integrates the appearance attributes, part geometry and their hierarchical structures in a unified framework. This new model significantly benefits from exploiting contextual information such as relative geometric position of parts as well as attribute correlation. For MSEE evaluation, the model has been trained to predict gender, glasses, hat, and color categories of upper and lower body, according to the evaluation framework.

- **Integration attributes information when tracking a person over time**. Previous work infers attributes from a single image. In a video, when a person is tracked over time, we can

further gather information over time. For example, we cannot infer whether the person wear a glass from a rear view, but if the person turns around, we can see his face in the other frame. Therefore, we developed a temporal-integration scheme to minimize the uncertainties arisen from ambiguous viewpoints or occlusion at individual frames.

3.3.3.1. Modelling and learning human attributes by attributed and-or graph

Based on the detected human figure and body parts as we showed in Section 3.3.2, our objective here is to further recognize attributes. The attribute model must be compatible and extend the and-or graph model for human pose in a principled way. Take gender as an example, both male and female have the same number of body parts, and the information must be integrated from subtle differences over all parts. Our model integrates the following aspects:

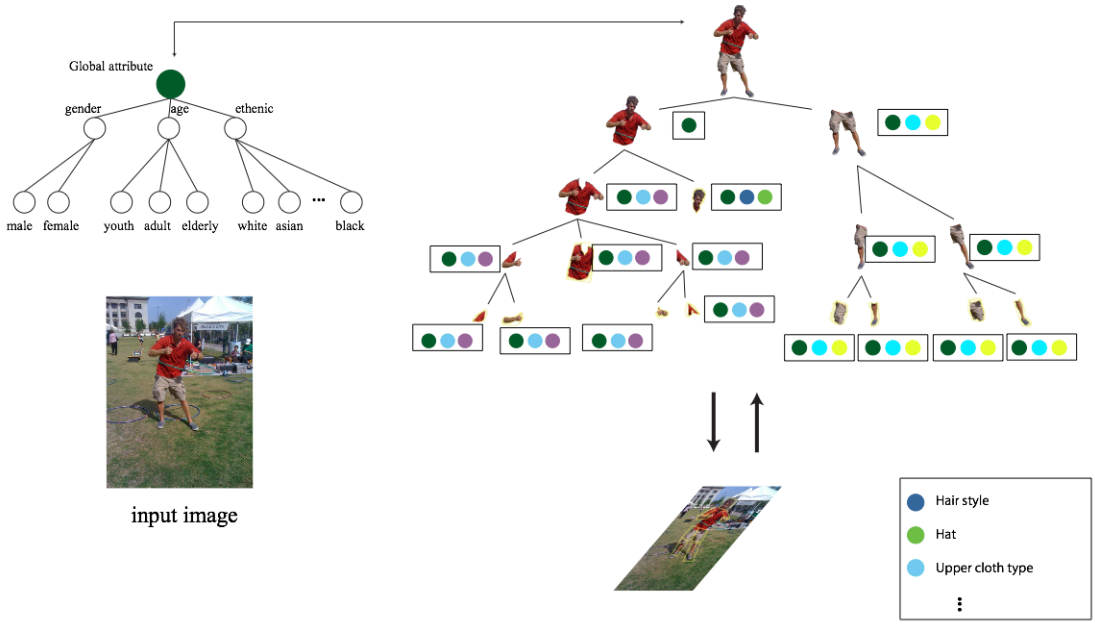



Figure 29. The attribute and-or graph represents the geometric decomposition and appearance types of parts. The global attributes are modeled as control variables which enforce the contextual constraints on the local attributes. The results are in an unpublished technical report.

- *Appearance cues*: male and female have different preference in their clothing appearance, such as texture and color;
- *Geometry clues*: pose, gesture and hair style (long, short, bald) have subtle difference,

  Not a single cue can deterministically decide the gender, we have to aggregate all features in the parse graph. Similar to the attributed and-or graph in scene parsing, the attributes are associated to node in different levels of the and-or graph.
- *Global attributes*: Gender, race, age, profession etc. are global appearance attributes associated with the root note (human). Certain distinct poses, such as crawling, crouching, are global pose attributes associated with the whole body.

- *Local attributes*: glass is associated with the face node, long/short hair is associated with head, jacket is with upper clothes, short/jean are with legs, and high heal is with feet.

  In the Attributed and-or graph, the attribute at a node A influences the branching probabilities of the Or-nodes under node A. Intuitively it acts as a controller. For example, a female has higher probability to have long hair, skirt, and high heels. All these terms are added to the probabilities in pose estimation and the algorithm solves for an optimal parse graph as the most probable and coherent interpretation of the pose, local and global attributes.

Figure 29 illustrates the model. The right side is the AoG for human body which decomposes the human into a set of articulated sub body part. Each part has a number of appearance templates as terminal nodes of the hierarchy. The attributes are shown as colored dots associated with different nodes in the hierarchy. The global attribute (dark green) is unfolded to show its own hierarchy.

3.3.3.2 Integrating attributes with human tracking module

Figure 30 shows some examples for why we need to integrate attribute information over time. By tracking the person in the video, we can overcome the problem of view, pose and occlusion. The sunglass is not visible in some frames but become visible when the person turns his head. The blue jean was occluded by the chair in some frame, and then become visible in other frames. Our model accumulated the scores for each attribute over the frames. The score is the log probability of the attributes in the parse graph. This step can significantly improve the performance. We need to infer attributes from multiple camera views and improve the performance for small person in distance and in low-resolution images.

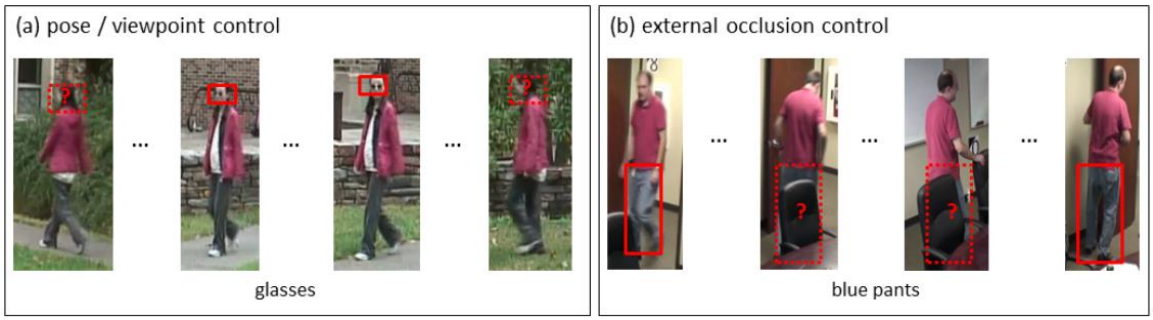



Figure 30. Because of (a) pose and viewpoint, and (b) occlusion by external object, the inference algorithm calculates attribute scores over given time frame, and it puts lower weights on uncertain attributes, which is marked by dotted rectangle, and higher weight on certain attributes.

### 3.3.4. Vehicle, parts, and attributes

Besides humans, vehicle is the second most important object category in videos. The MSEE task for vehicle include *car detection*, *part localization*, and *attributes (color).* Our work has been mostly focused on sedan (and less on other sub-categories, like bus, truck etc.). Cars are rigid and thus have less variations than humans, but a new challenge is that cars have far more severe

occlusions than humans, as they are often parked in high density to save space. Thus our efforts in Phase II are aimed to tackle the occlusion issue. During this period, we have published 2 more papers on car detection and part localization. Since we have reported the 3D And-Or graph for car parsing in previous report, here we only present the recent work on cars in our two papers.

3.3.4.1. Modeling occlusions between vehicles

Figure 31. Examples of cars in our street and parking dataset released to the public.

To handle car occlusion problems, we have collected a large dataset of cars on street and parking lots for training and testing. Figure 31 shows some examples. Occlusion can severely deteriorate the detection performance because the occluded parts provides false features to match with the compositional templates. To overcome this problem, we extend the car model by explicitly representing the occlusion patterns. Suppose a car has 17 parts, each part may be occluded, this leads to $2^{17}$ possible configurations. However, in practice the most frequent occlusion patterns are far less. Figure 32 illustrates the typical car-to-car occlusion configurations which depend on the view angles and the other cars around. To model these configurations, we developed a compositional and-or graph, which is shown on the right of Figure 18, to account for the combinatorial effects and regularity.

Since it is hard to take images for various occlusion configuration, we use 3D CAD models of cars and divide each model into 17 semantic parts. By controlling four factors: car types, orientations, relative positions of parked cars, and camera viewpoints, we generate a huge set of occlusion configurations. Figure 18 (left panel) shows some of the configurations. In this simulated data, we know exactly which parts are occluded. We then use these occlusion maps to train an and-or graph for occlusion (see Figure 32, right panel). This And-or graph defines what type of combination is plausible. A parse graph from this and-or graph represents all the visible parts and only the visible parts are matched to the templates to resolve the occlusion problem.

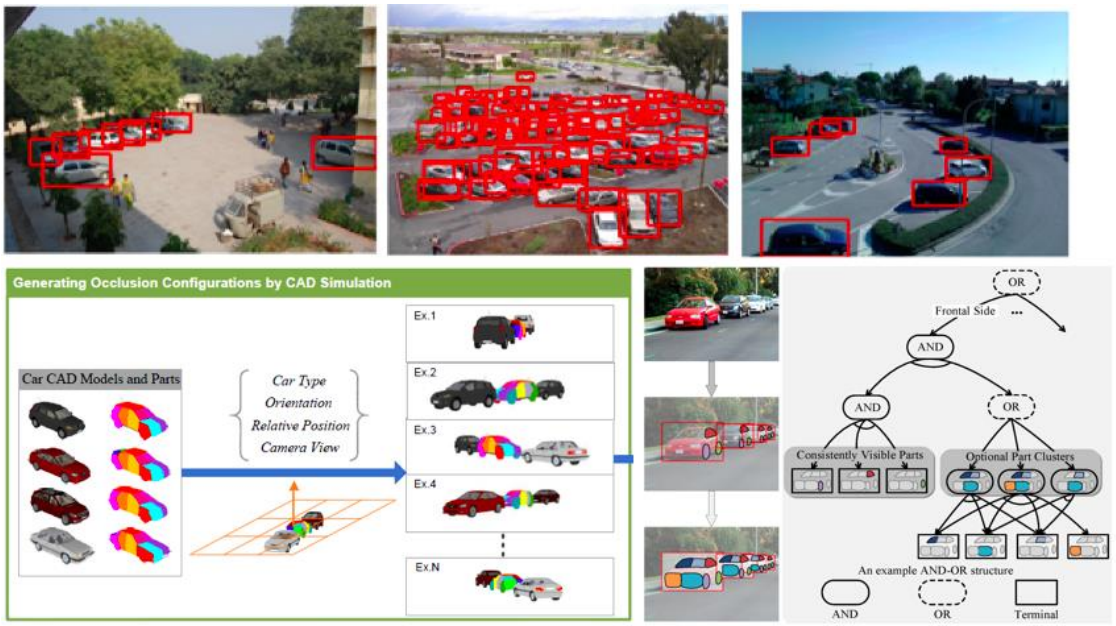



Figure 32. The and-Or graph for modeling object occlusions. It organizes object parts into consistently visible parts in And-node and optional part clusters in Or-nodes. A parse graph is composed of the plausibly visible parts for an occluded car.

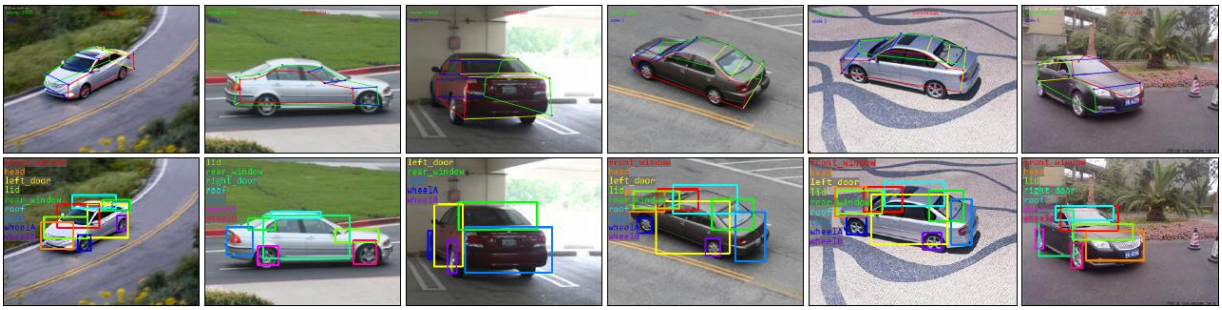


Figure 33 Detecting car, estimating 3D pose, and localizing car parts on a public dataset.

| Methods | Easy | Moderate | Hard |
|---|---|---|---|
| mBoW[1] | 36.02% | 23.76% | 18.44% |
| LSVM-MDPM-us[2] | 66.53% | 55.42% | 41.04% |
| LSVM-MDPM-sv[2,3] | 68.02% | 56.48% | 44.18% |
| MDPM-un-BB[2] | 71.19% | 62.16% | 48.43% |
| OC-DPM[4] | 74.94% | 65.95% | 53.86% |
| DPM-trained-boli | 77.24% | 56.02% | 43.14% |
| Car_Comb_AOG | **80.26%** | **67.03%** | **55.60%** |

Figure 34. Detection comparison on the KITTI vision benchmark according to their protocol. Our results ( Car_Comb_AOG) are reported in the last row in red. Note that we use half of the training set, while other tested methods in the benchmark use more training data.

We obtain the state-of-the-art performance on the popular PASCAL VOC 2007 car dataset and a very large public benchmark --- the KITTI Vision Benchmark. We also tested against the

PASCAL VOC benchmark on cars and beat the top performers, especially we have much higher precision on localizing the car parts. Figure 33 show the results on part localization. SIG queries test car parts like door, roof, bumper etc. The comparison results are shown in Figure 34.

### 3.3.5. Functional objects: furniture

The third big category of objects in daily videos is the functional objects, including chairs, tables, desks, doors, water bubblers, cabinets, etc. Unlike humans and vehicle which are defined and detectable by their geometry (shapes, poses, views) and appearance (edges, textures, colors, and shading), the concept of furniture is defined by functionality, i.e. how it is used by humans in the scene. For example, a chair or sofa could have many, literally endless, designs of geometry and appearance, therefore it is infeasible to define a model and learn it from training example. We developed a new method that reasons the functions of an object by imagine a human figure in the scene. This work was published at CVPR 2013 and a long version is accepted by Int'l J. of Computer Vision.

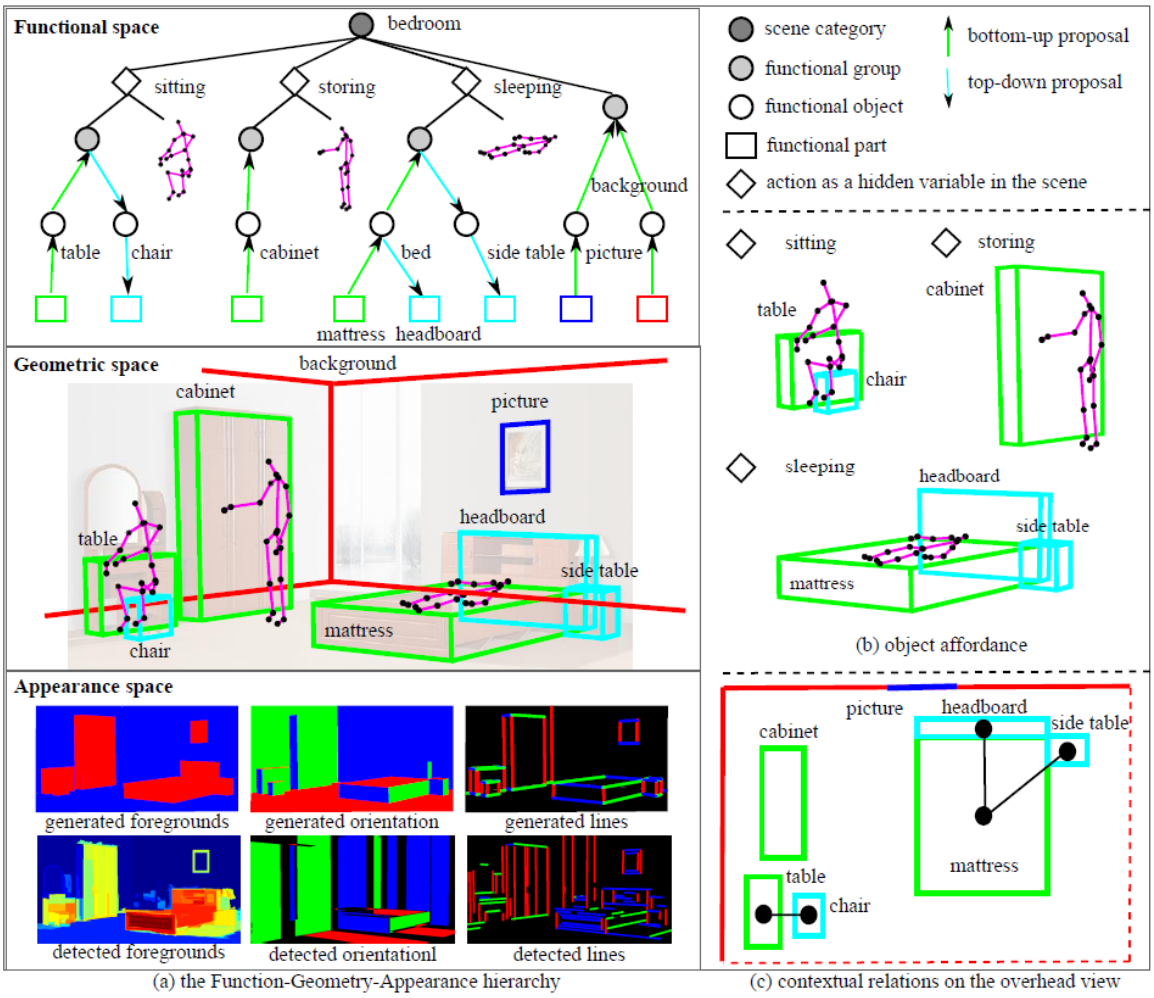
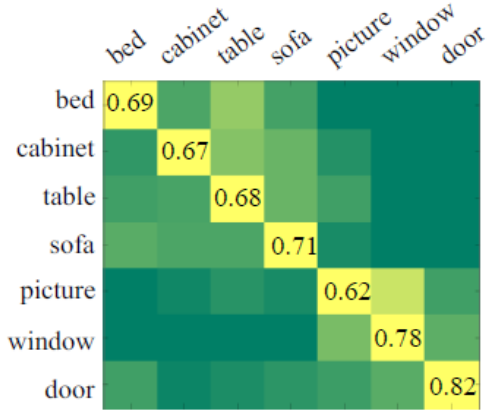



Figure 35. Understanding scenes and functional objects by reasoning plausible human actions.

Figure 35 illustrates the method for recognizing objects in the bed room. We extend the and-or graph (including the appearance and geometric size) by adding one more layer on the top --- the functional space. This layer represents a number of typical scenes, like bedroom, kitchen, and each scene has a few actions that people usually do. For example, a bedroom serves three typical actions: sitting (dressing hair, making up face), storing/taking clothes from cabinet, or lying in bed. Each of these actions is further decomposed into a number geometric relations between the human pose and the functional objects.


Figure 36. Parsing results include cubic objects (green cuboids are detected by bottom-up step, and cyan cuboids are detected by top-down prediction), planar objects (blue rectangles), background layout (red box). The parse tree is shown to the right of each image.




Confusion matrix for the 7 classes of furniture.

We have built the models from three aspects:

- Collecting 3D human poses (skeletons) using Kinect cameras for typical human actions (sitting, lying, walking etc., see the action section 3.4.3);
- Pooling statistics of the dimensions of furniture from the 3D warehouse of Ikea furniture which are available online;
- Learning the 3D relations between human pose, body parts with the functional objects using Kinect cameras (see Section 3.4.3).

  Based on these statistics and models, we developed an integrated method for simultaneously parsing the scene (from 2d images) and reasoning the object functions by imaging possible human actions. Figure 36 shows some of the results for indoor scene parsing where the parse trees are shown on the right side of the 2D images. And the Table in the previous page is the confusion matrices between the 7 classes of furniture. Obviously there are still room for improvements.
- Using video and thus observed human actions (not imagined) to reason the objects, this was demonstrated in a restricted setting in our STC-parsing demo video;

### 3.3.6. Other object categories

Besides humans, cars, and furniture, there are other object categories. In the SIG video, many small objects can often detected through their motion: throwing a ball, leaving a box or backpack. In Phase II, we have developed a general and-or graph model and train it using discriminative methods. This method beat the state-of-the-art on the PASCAL VOL 20 object categories and reported at CVPR13, which is under extension to a journal paper. Table 9 below shows the current detection performance. AOG Disc. is our method and is compared against the deformable parts model (DPM release 5).

| Table 3 | | aero | bike | boat | bttle | bus | car | mbik | train | bird | cat | cow | dog | hrse | sheep | pers | plant | chair | tble | sofa | tv | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 20 07 | DPM r5 | 32.4 | 57.7 | 15.7 | 25.3 | 51.3 | 54.2 | 47.5 | 41.3 | 10.7 | 17.9 | 24.0 | 11.6 | 55.6 | 22.6 | 43.5 | 14.5 | 21.0 | 25.7 | 34.2 | 44.2 | 32.5 |
| | AOG discr | **35.3** | **60.2** | **16.6** | **29.5** | **53.0** | **57.1** | **49.9** | **48.5** | **11.0** | **23.0** | **27.7** | **13.1** | **58.9** | 22.4 | 41.4 | **16.0** | **22.9** | **28.6** | **37.2** | 42.4 | **34.7** |
| 2010 | DPM r5 | 42.9 | 47.2 | 11.1 | 26.3 | **48.4** | 40.2 | 44.0 | 39 | 10.3 | **22.9** | 22.9 | 19.9 | **41.5** | 28.3 | **41.0** | 7.6 | 17.0 | 10.2 | 18.2 | **32.9** | 28.6 |
| | AOG discr. | **44.6** | **48.5** | **12.9** | 26.3 | 47.5 | **41.6** | **45.3** | 39 | **10.8** | 21.6 | **23.6** | **22.9** | **40.9** | **30.4** | 37.9 | **9.6** | **17.3** | **11.5** | **25.3** | 31.2 | **29.4** |

**Comparison on VOC 2007 and 2010** benchmarks 20-class between Deformable Parts Models (release version 5

Figure 37 shows some successful results of detecting 20 classes of objects and their parts on the VOC dataset. There are still room for improvements in Phase III for this module:

- Enhancing the features objects;
- Integrating object detection and recognition with human actions in video;
- Modeling better and richer scene context model.

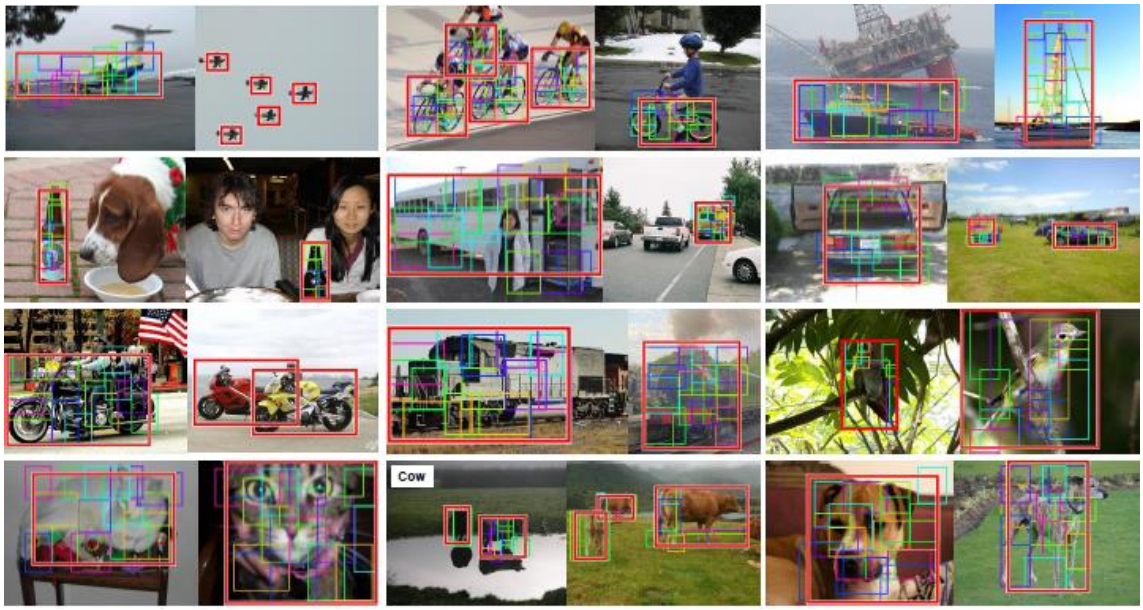



Figure 37. Detecting objects and localizing parts on the PASCAL VOC dataset.

### 3.4. Temporal parsing

Temporal parsing includes a ranges of challenging tasks in videos across a network of cameras.

- Tracking objects in long time across cameras in 2D image and 3D world coordinate;
- Human action recognition from poses across varying views;
- Human action recognition based on scene and contextual objects;
- Activities involving humans and vehicle interactions;
- Reasoning human intents, trajectories and events prediction;
- Event recognition through event grammar.

#### 3.4.1. Tracking objects in long videos

To track multiple objects human, vehicles, animals, balls, packages, bags, and other objects in video, we have developed two tracking algorithms corresponding to the following two task settings which are commonly studied in the computer vision literature.

- *Off-line Tracking*. Given a video clip, the algorithm computes the trajectories of all objects using all frames in the video. It is called offline, because the algorithm can reason backwards and forewords in time from each frame.
- *On-line Tracking*. Given a bounding box for an object at the initial frame, the algorithm must output the current position at frame t without using the frames after time t.

  The tracking algorithm provides trajectories of objects for other applications such as attributes recognition, action detection and recognition.

3.4.1.1 Offline Tracking

In the offline tracking task, we first run the background modeling module to detect the moving objects in the foreground, and then on the foreground regions we run various object detection modules --- detecting humans, cars, animals, etc. The unrecognized foreground moving blobs are often the "other objects", such as bags, pushed chairs, and unfamiliar objects. These detection modules produce a list of candidates. Figure 38 shows an illustration where each candidate at a time frame (horizontal axis) is shown as a yellow circle. The remaining task is posed as a data association problem and can be solved by dynamic programming.

One complication is to handle the missing or noisy detections, especially when multiple object cross each other. We introduce a prior model to enforce trajectory continuity and smoothness and to link short fragments into long ones.

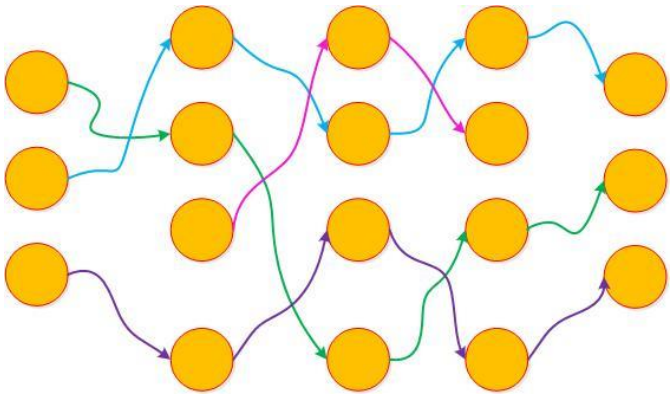


Figure 38. The horizontal axis is time frame. For each frame a few object candidates are detected and shown as yellow circles in each column. By matching and tracing the candidates over frames, the algorithm outputs a number of trajectories in colored curves.

3.4.1.2 Online Tracking and learning

In the online tracking task, the bounding box of the object of interest is initialized at the first frame, say by other modules, or by the user. The tracking algorithm is supposed to output the trajectory of the object in the successive video sequence on the fly, i.e. without delay. In online tracking task, there are six known challenges:

- Occlusion, e.g. a person sits down behind a table;
- scale change, e.g. a car drives from far to near;
- illumination change, e.g. an object moves from sunlight to shadow;
- pose change, e.g. a person standing begins to crawl on grass;
- appearance change, e.g. a person takes of clothes or turn around; and
- confusion with similar objects, e.g. basketball players in the same team.

We developed a framework for simultaneously solving three sub-tasks: tracking, learning and parsing, using a hierarchical and-or graph representation. The and-or graph represents the

variations of the object under tracking, and is updated online to learn the geometry and appearance of the object. This online learning and tracking is particularly important when the object is not pre-trained and unfamiliar. Figure 39 shows the AOGTracker framework.

| | AOG | Struck | CXT | VTD | VTS | OAB | CPF | LSK | FRAG | MIL |
|---|---|---|---|---|---|---|---|---|---|---|
| Prec. | **0.851** | 0.773 | 0.658 | 0.650 | 0.645 | 0.604 | 0.599 | 0.589 | 0.582 | 0.574 |
| Suc. | **0.748** | 0.694 | 0.579 | 0.583 | 0.581 | 0.540 | 0.502 | 0.556 | 0.530 | 0.496 |

**Table 10**. Performance comparison of the top 10 trackers evaluated on the 50-video benchmark.

We have compared our method (AOG) for online tracking against other methods in a public dataset, which includes 100 challenging videos. The result shows our algorithm outperforms other state-of-art methods significantly. The numeric numbers are the precision of the bounding box compared with the ground truth human annotated by humans.

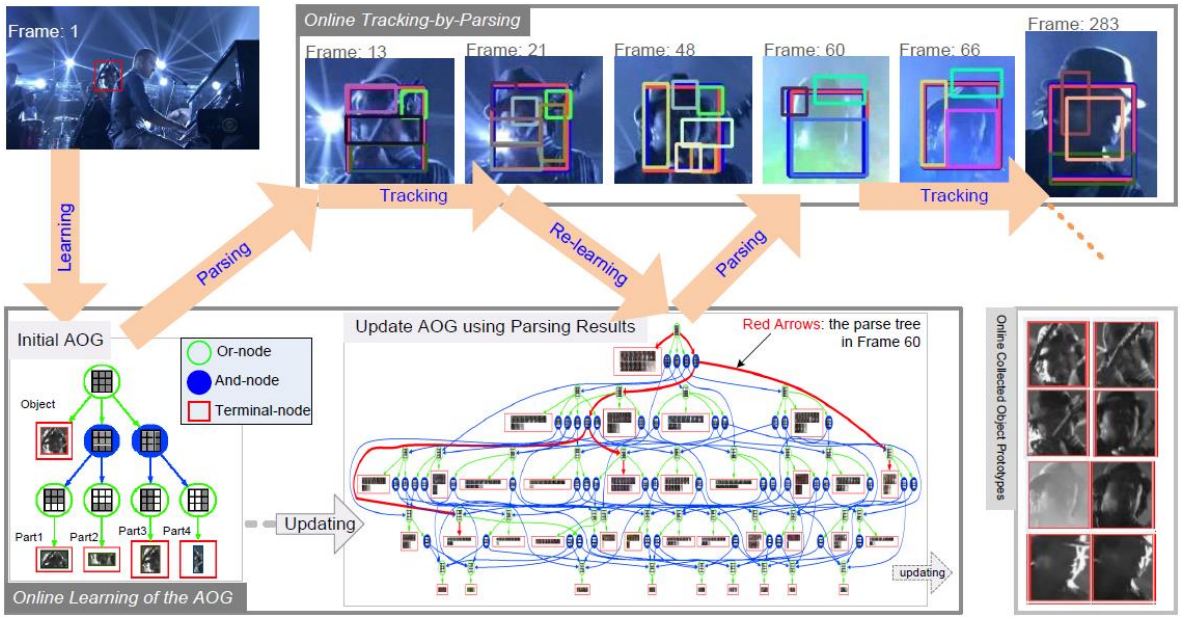



Figure 39. Illustration of our online tracking-learning-parsing framework based on the and-or graph representation. As the object is tracked over time, the algorithm learns an AoG representation for its geometry and appearance including the changing pose, view and scale etc.

### 3.4.2. Human action recognition across multi-views

In the computer vision literature of video-based action recognition, most existing methods recognize actions from the view that is more or less the same as the views in the training videos. Their general limitation is the unpredictable performance in situations where the actions need to be recognized from a novel camera view. Our research is focused on recognizing cross-view actions, i.e., actions from unseen novel views.

We approach this problem from a new perspective: creating a generative cross-view video action representation by exploiting the compositional structure in spatio-temporal patterns and geometrical relations among views. This model is called Multiview Spatio-Temporal And-Or-Graph, or MST-AOG. This new MST-AOG model has the following advantages;

- It is a compact but expressive multi-view action representation that unifies the modeling of geometry, appearance and motion.
- Once trained, this MST-AOG model only needs 2D video inputs to recognize actions from novel views, and no 3D inputs are needed.
- To train this MST-AOG model, we provide new and effective methods to learn its parameters, as well as mining its structure to enable effective part sharing.

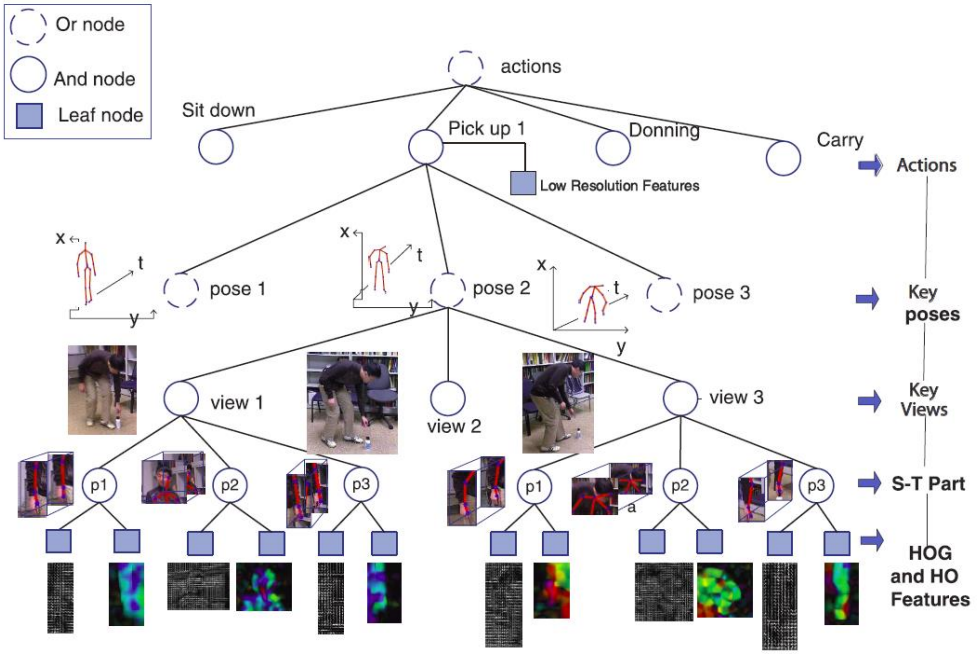



Figure 40. The MST-AOG action representation. The geometrical relationship of the parts in different views are modeled jointly by projecting the 3D poses into the given view. The discriminative parts are automatically learned and shared for all the actions.

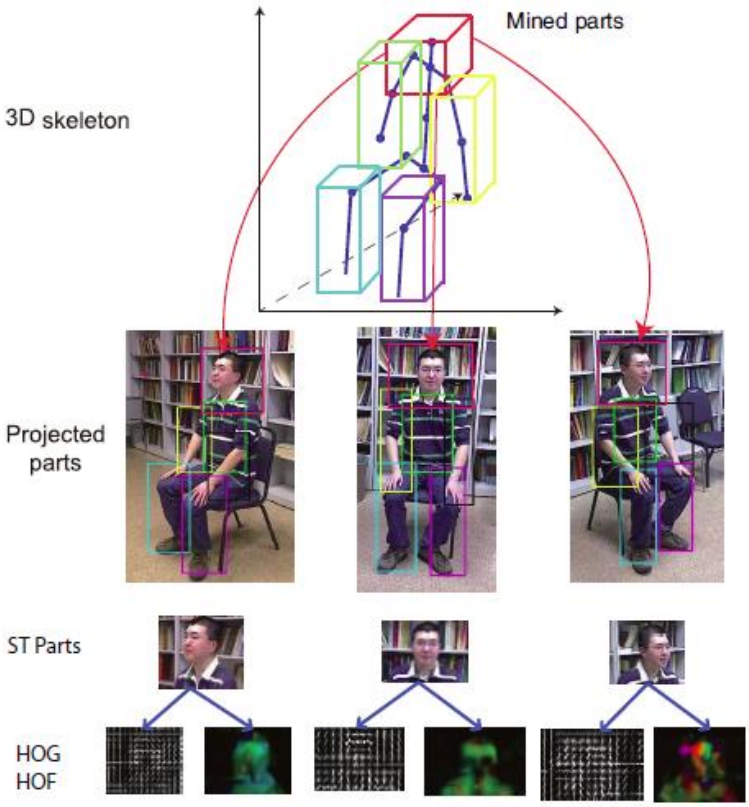



Figure 41. A 3D skeleton information is collected from Kinect cameras and is used in training, but not used for cross-view action recognition. The projection of the 3D poses enables explicit modeling of the 2D views. Our model uses a set of discrete views in training to interpolate arbitrary novel views in testing.

Figure 40 illustrate a portion of the MST-AOG model, which jointly models the following information about the actions: appearance, motion, geometric relationship, and low-resolution features. An important property of the MST-AOG is that it shares parts across different views by learning a 3D model, so that it can generate projections to new views.

We train the 3D action model using a dataset captured by depth sensors. Each action is acted by 9 people in an indoor scene and simultaneously captured by 3 Kinect cameras from different angles. Then we registered the 3d skeletons from the 3 cameras and produce a 3D skeleton as ground truth. This skeleton produces annotation of the pose and body parts. Therefore, we can project the geometry and dynamics to arbitrary views, as it is shown in Figure 41.

We create a new dataset, the multi-view 3D event dataset, which contains RGB, depth and human skeleton data captured simultaneously by the three Kinect cameras. We compare the proposed MST-AOG method with the state-of-the-art cross-view action recognition methods under three settings: cross-subject setting, cross-view setting, and cross-environment setting. The experimental results are summarized in Table 11 below. The proposed algorithm achieves the best performance under all three settings. Moreover, the proposed method is very robust under the cross-view setting. Figure 42 shows the recognition rate of the various action categories.

| Method | Cross-Subject | Cross-View | Cross-Environment |
|---|---|---|---|
| Virtual View | 0.507 | 0.478 | 0.274 |
| Hankelet | 0.542 | 0.452 | 0.286 |
| Action Bank | 0.246 | 0.176 | N/A |
| Poselet | 0.549 | 0.245 | 0.485 |
| Mixture of DPM | 0.748 | 0.461 | 0.688 |
| MST-AOG w/o Low-res | 0.789 | 0.653 | 0.719 |
| MST-AOG w Low-res | **0.816** | **0.733** | **0.793** |

Table 11: Recognition accuracy on Multiview-3D dataset

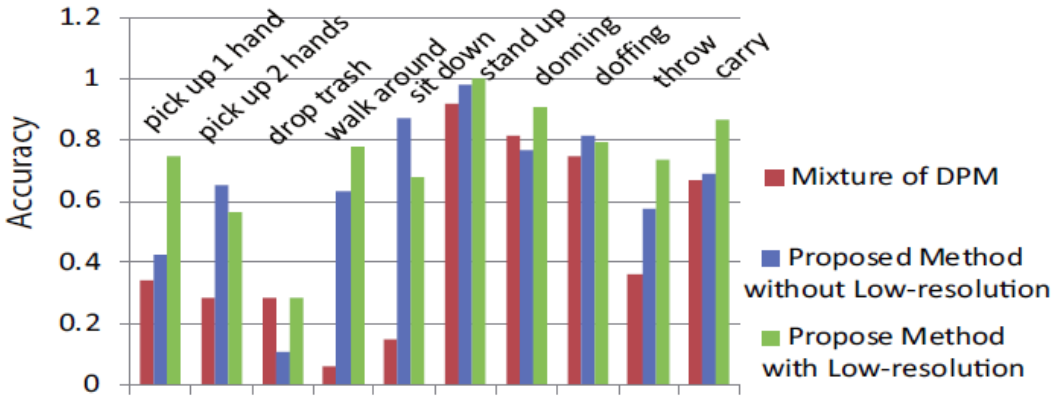



Figure 42. The recognition accuracy under cross-view setting.

### 3.4.3. Human action by scene context

Scene context plays a very important role in improving action recognition. Some actions, like picking, waving, clapping, and boxing, can be recognized by the poses as we discussed in Section 3.4.2, but many actions are not. For example, sitting on the chair, writing on a white board, drinking, reading, and writing. In Figure 43, reading and writing can happen by the side of table and taking box happens near table. This concept has been discussed in Figure 35 where we used imagined human actions to recognize scene and furniture. Here the information flows in

the opposite direction: from scene and furniture to action recognition. This is a characteristics of the MSEE project --- the joint representation supports the joint inferences so that the different dimensions: scene, objects, actions and event etc. can help each other to form joint interpretations.

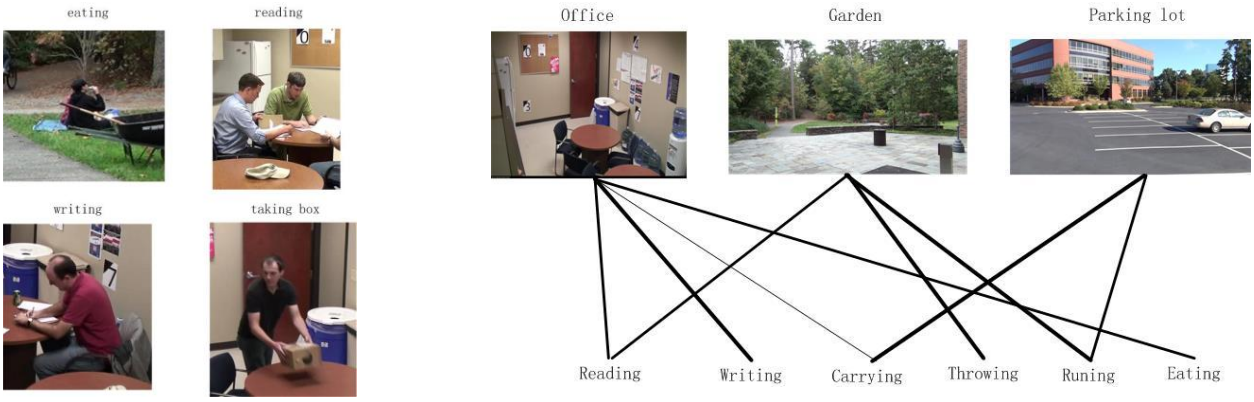



Figure 43. Some actions need scene context to assist the recognition. Actions have different probabilities to occur in certain scenes.

We utilize two types of context information to help action recognition.

- One is the scene context which can be seen as the global feature which can provide different weight for different actions. The weight can be seen as the probability that certain action can happen in this scene. For example, some actions have high chance to happen in in-door scene such as taking box, putting box, reading and writing, while other actions always happen in out-door scene such as running, crawling, eating and throwing.

- Another is object context. The surrounding object of different actions can provide critical information to action recognition. For example, walking and running usually happen on the ground, taking box happens near the table, eating can happen on the grass (picnic) or near the table, and loading happens near the vehicle etc.

In a separate project, we have learned the 3D geometrical relations between human pose and body parts with contextual objects in the scenes. Results are reported in two ICCV papers [7, 9]. We collected the Kinect RGBD videos for some action categories as we did for the multi-view action recognition. Figure 44 shows some of the typical 3D relations. Each objects, like chair, desk, book, monitor, button (or switch) on the wall etc. are approximated by 3D boxes. The 3D relations are modeled by multi-variety Gaussians and also integrated over time. We call this model 4DHOI (4 dimensional human-object interaction). We used this relations to assist action recognition, contextual object detection (furniture). Figure 45 shows some comparison results for functional object detection using the 4DHOI (Fig.45.(d)) against other methods: HOG + SVM, and RDH.

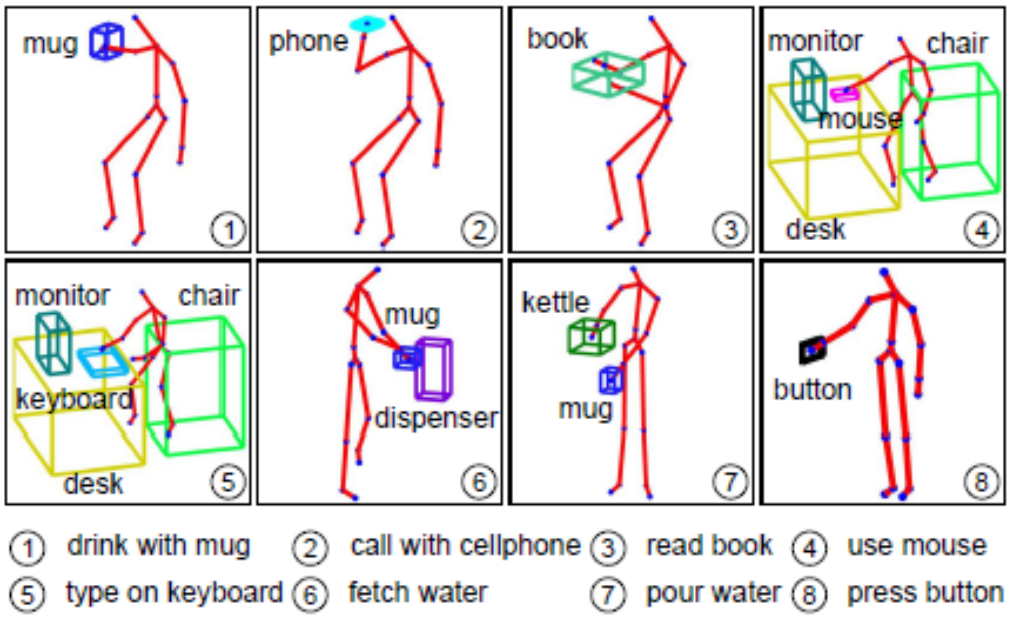



Figure 44. Learning the 3D geometric relations between human poses in action and the contextual objects, such as tables, chairs, monitor, mugs, boxes, switch on the wall etc.

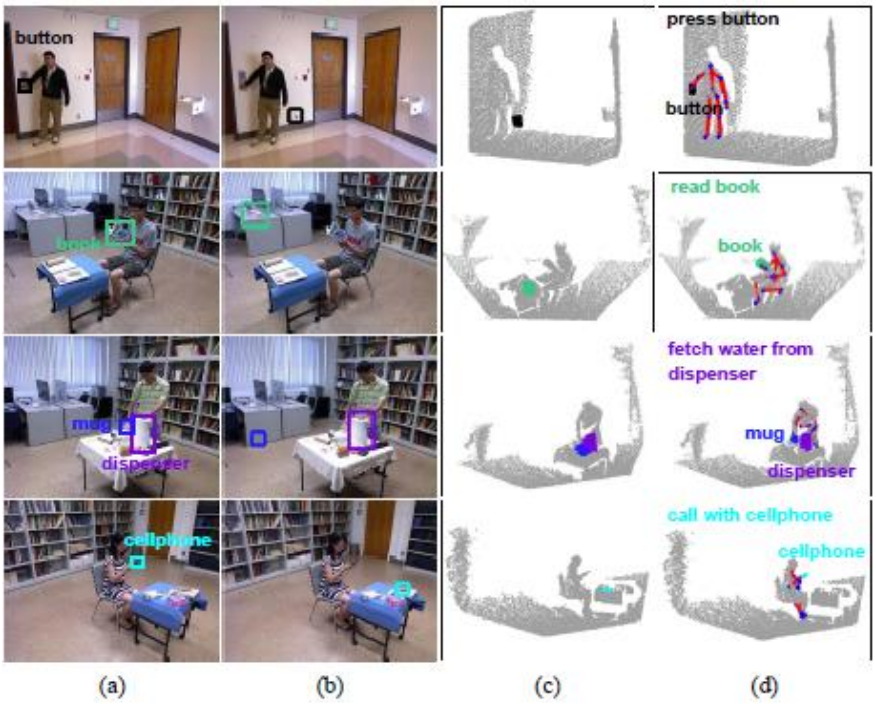



Figure 45. Object recognition and localization. (a) Ground truth. (b) HOG. (c) RDH. (d) Our 4DHOI. The results of RDH and 4DHOI are visualized by projecting the areas on the depth images into the 3D point cloud.

### 3.4.4. Activities involving human, vehicles and regions

Besides the single human actions, the MSEE evaluation framework includes a number of activities which are defined by the motion patterns and the interactions between humans, bicycles and vehicles. For example, *starting, moving, stopping, stationary, turning, turning-right, turning-left, u-turn, driving, crossing, mounting, dismounting, loading, unloading, together, same-motion, opposite-motion, following, passing, talking, eating* (picnic with multi-person). Recognizing these activities relies heavily on the tracking of the objects in video, and our method overcome the following main difficulties.

- The motion projected in the 2D image frame lost critical information, we transform the coordinates in 2D video in the 3D world coordinates. To do so, we track the head of humans (as feet are often occluded), and estimate the trajectory of the human in the 3D world coordinate using the projection matrix (camera) and the size of the head.
- Some small objects cannot be reliably detected in video. For example, when people load some relative small object into the car, this object is difficult to detect without knowing what it is in advance. Therefore, we resolve this problem by high level reasoning based on their spatial and temporal interaction.

In the following, we show four typical activities which are defined by a unary or binary relations based on their trajectories and speed.

Figure 46 shows the first example for how we compute the velocity of a person riding a bike. The 2D coordinates in the video are transformed to world coordinates on the ground (longitude, latitude) from which we estimate the ground speed (see the right panel). The trajectory is then divided into three motion statuses: *Stationary, Starting, and Moving.*

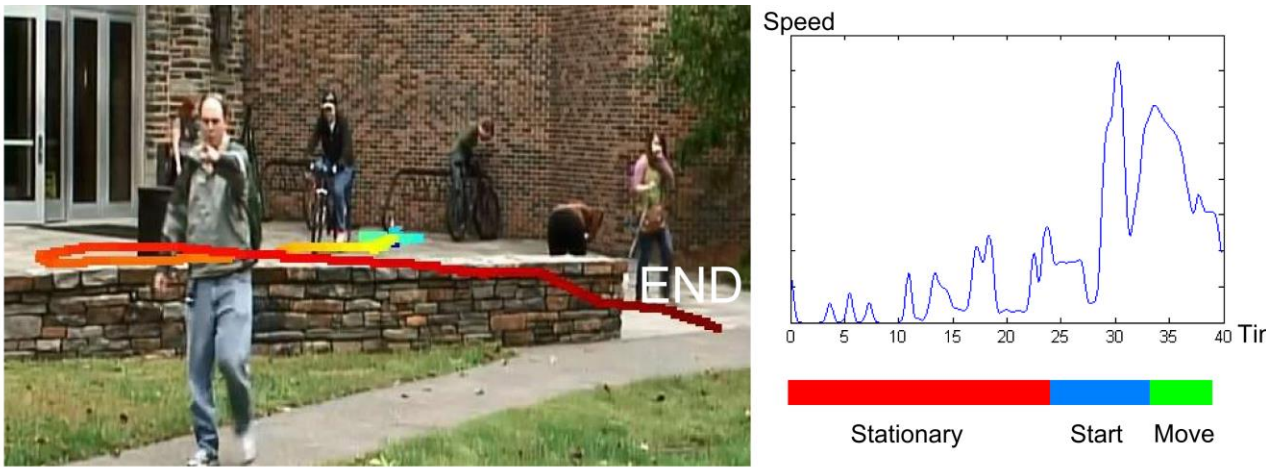



Figure 46. The left panel shows a person on a bicycle with a colored trajectory (color represents time) in a garden video. The right part shows the speed of the person at each moment based on the trajectory in the left video. Note that colors on the right do not correspond to the color of the trajectory on the left.

Figure 47 is an example for estimating the turning behaviors: *turning, turning-left, turing-right* and *u-turn.* As we can see, turning is a gradual and smooth behavior over time and it should be

estimated based on a certain time window or time scale. Our answers to such queries are probabilistic and parameters are learned to match human intuition of what means turning.

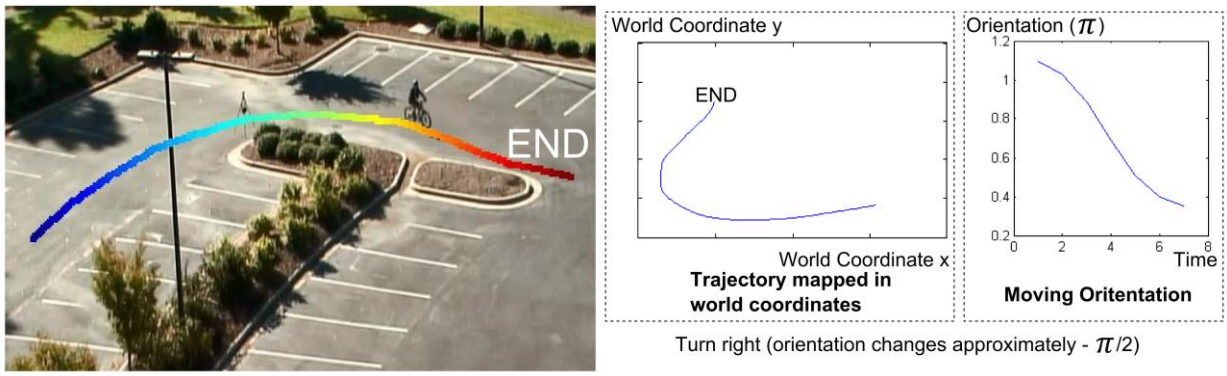



Figure 47. (Left) colored trajectory (color represents time) of a biker in a parking lot. (Middle) the trajectory in world coordinates (longitude, altitude). (Right) moving direction (angle) of the biker.

Figure 48 shows examples of binary behaviors between two bikers: *behind*, *following* and *passing*. At any moment, we estimate their relative position based on their moving direction. Then these behaviors are recognized by detecting the changes of their relative positions in temporal sequences.

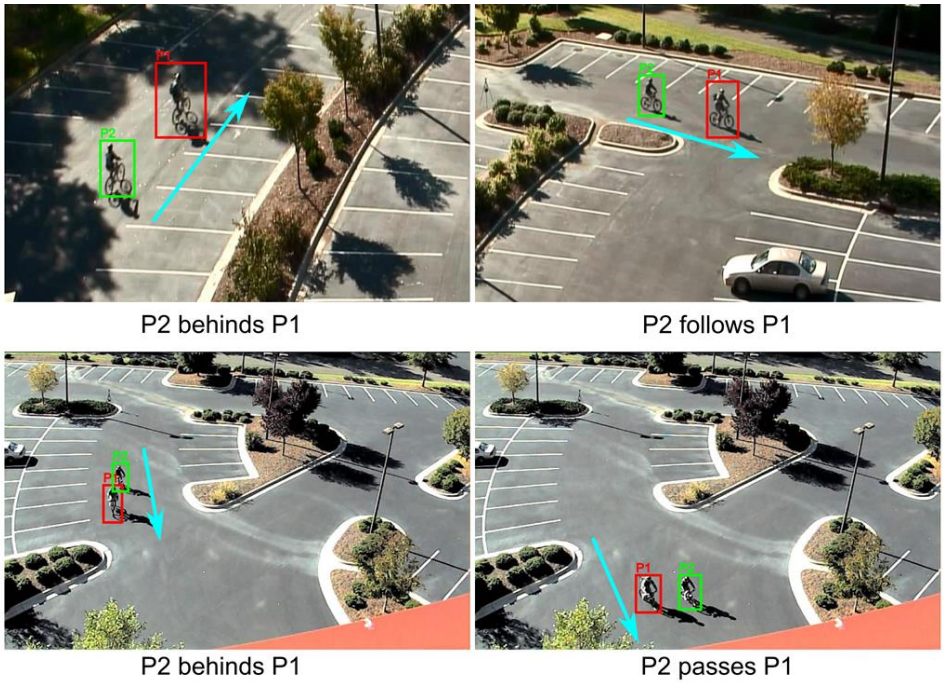



Figure 48. The top row shows a video that person P2 follows P1. The bottom row shows a video that person P2 passes P1.

Figure 49 is an example of behavior among multiple person/object: *loading* and *unloading*. Here two people loaded something to the truck.

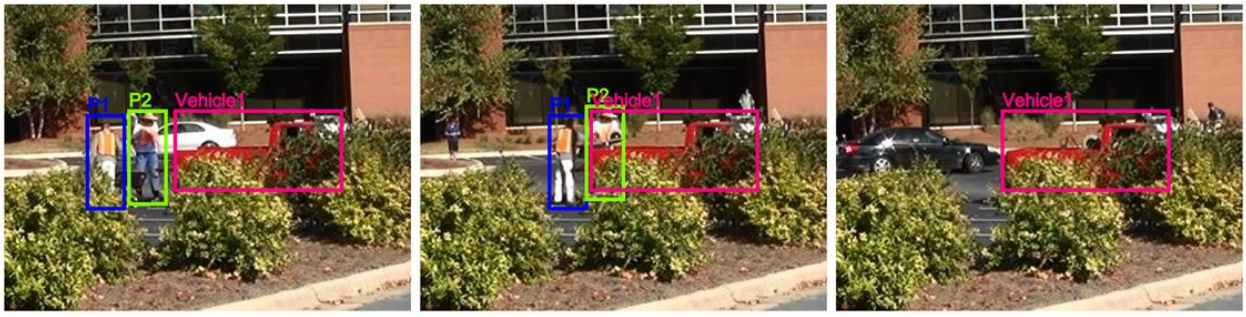



Figure 49. We infer that the two people are loading something instead of unloading something based on their relative positions with the car in temporal sequences.

### 3.4.5. Human intents, trajectories and events prediction

In Phase II, we also developed a module analyzing on the UCLA courtyard activities and group activities in the aerial video collected by a UAV over Malibu state park. In such videos, the human and objects are too small to be recognized, but human trajectories can be computed (imperfect). And a good thing is that the trajectories are on the ground coordinate thank to the top-view. Our objective is two-folded:

- Inferring, through reasoning, the functional objects in the scene, such as, chairs, trashcan, BBQ stance, vending machine, food truck, picnic table etc. These objects provide certain functions and attract people.
- Inferring the human intents, such as hunger, thirst etc. and thus predict their trajectories and events.

For example, in Figure 50, when a person is walking in the scene, we can estimate the probabilities for the person to get food from a food truck, entering a building, or get a drink from the vending machine etc. The probability is updated over time based on the observed trajectory. Based on the probabilities, we can also predict the possible movement.

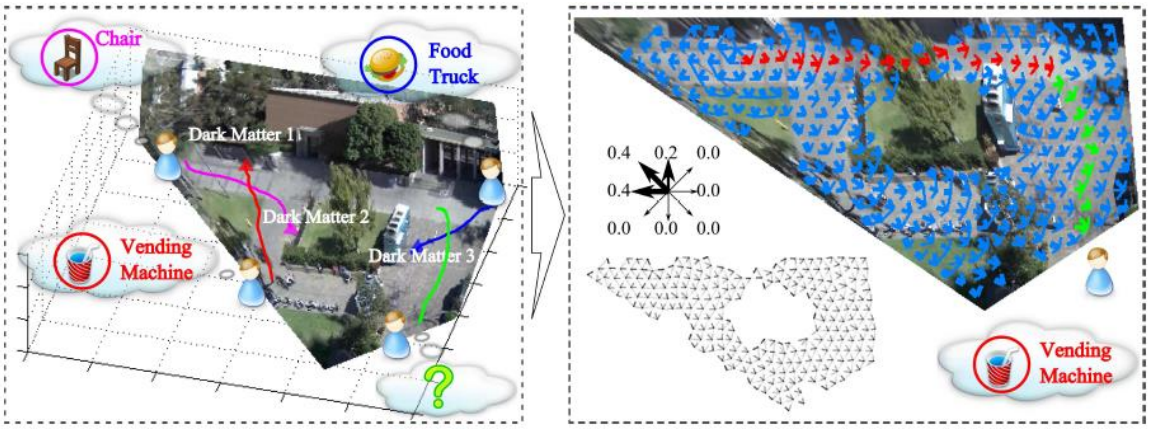



Figure 50. (Left) A courtyard scene with multiple attractions. (Right) the prediction of trajectory (red) for a person based on the observed trajectory (green).

Our method extends the Lagrangian mechanics in physics and is published in ICCV [2]. Each functional object in the scene is modeled as a source that emit a field of attraction (or repulsion) in the scene. Thus the scene has many layers of fields. When a person is triggered by a certain intent. His/her movement follows the field emitted by that corresponding object, just like a particle moving in the field of gravity or electro-magnetism. The motion equation can be derived following the Lagrangian mechanics.

- *In a learning mode*, our algorithm observes the motion of many trajectories and thus can learn the type of fields for different categories of functional objects and human people interact with them. For example, queueing in front of a food truck.
- *In an inference model*, our algorithm estimate the human intents and predicts the trajectories.

Our task is different from physics in several aspects, which make the problem interesting:

- Humans can change mind in the middle of the execution a certain action;
- Humans who are familiar with the scene have a "global map" and thus can plan for a globally optimal path, instead of a greedy path;
- Humans may attract each other and thus become a moving attraction field on their own.

Figure 51 shows the results on four scenes. We show how we estimate the attractions (sources S) and obstacles (Constraint C, such as grass, tree where people have to avoid) in a learning mode (see the middle panel), and the trajectory predictions in terms of probability (see the right).

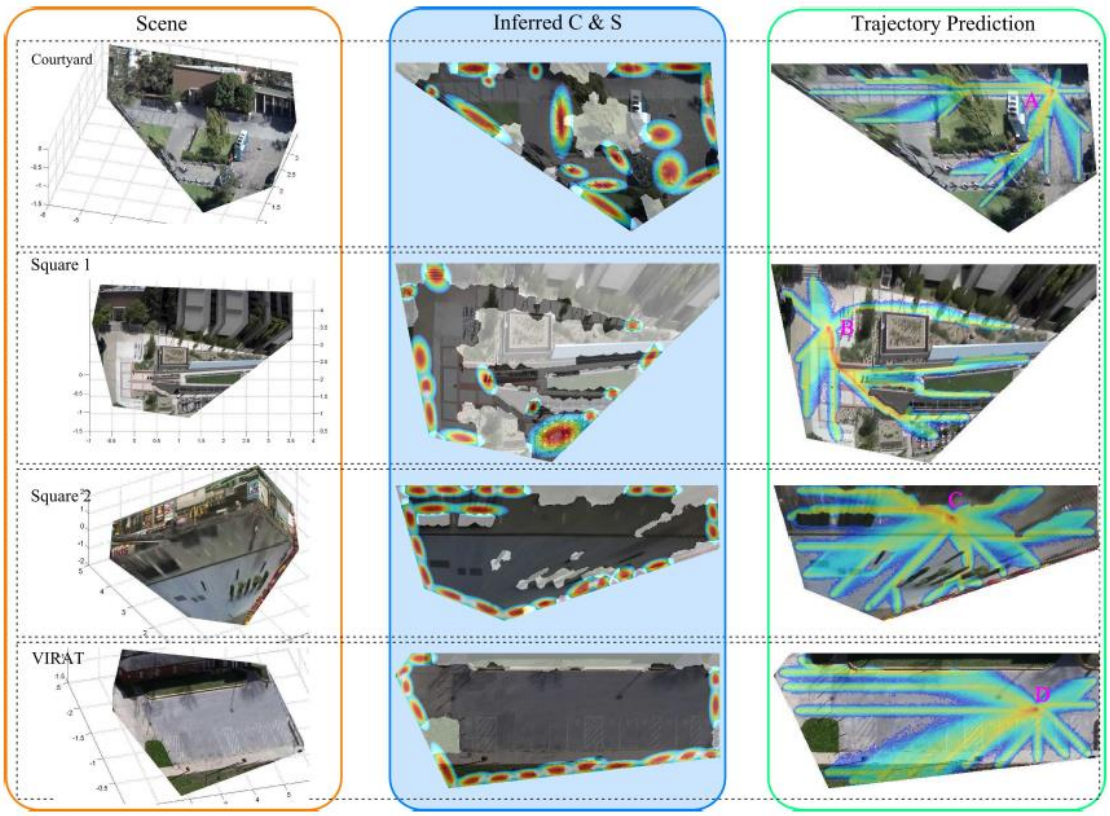



Figure 51. Qualitative experiment results for 4 scenes. Each row is one scene. The 1st column is the reconstructed 3D surfaces of each scene. The 2nd column is the estimated layout of obstacles (the white masks) and attractions (position are estimated by ellipses). The 3rd column is an example of trajectory prediction, we predict the future trajectory for a particular agent at some position (A, B, C, D) in the scene towards the potential attractions, the warm and cold color represent high and low probabilities of visiting that attraction respectively. From [2].

### 3.5. Query answering

The Query Engine Module is implemented by our team members in IAI for the MSEE evaluation. We have developed two versions of the query engine:

- One version answers the questions on what, who, where, when and why; this was illustrated in our demos.
- The other version answers the yes/no binary questions designed by SIG according to the restricted Turing tests.

Both query versions support the storylines and probabilistic answers (with uncertainty).

The MSEE system is designed to consume Scene Observation Collections (SOCs), which include sensor data (with metadata) from multiple sensor sources along with scene descriptive text recorded in an area of responsibility (AOR). Based on these inputs, the MSEE system performs joint spatial and temporal parsing and reasoning, as we discussed in previous Sections.

The product is a *Joint Parse Graph* that describes the predicates (labels, states, and relations) detected in the AOR.

To evaluate the accuracy, the MSEE system provides an interface for a user or an evaluation engine (assessor) to ask true/false queries. For example, it might ask whether there are at least two feet in the reception room at 15:01:43 on 9-4-2013. The system then computes the most probable answer to the query along with a confidence score between 0 and 1, which is intended to estimate the probability that the answer is correct.

A Query Engine module is being developed to retrieve answers for the queries issued by the MSEE Assessor. This module can currently derive the answers to many queries based on information in the Joint Parse Graph.

The architecture of the Query Engine module is shown in Figure 52. It consists of four main components:

- A **Query Translator** that convert query from an XML form to a SPARQL form.
- **Jena Query Engine** that interfaces with the JENA SPARQL API to execute the query on the RDF data, and executes special JAVA functions to handle spatial-temporal conditions.
- **A Probabilistic Post-process module** that integrates results from multiple parsed graph interpretations.
- **An Answer Interface** that outputs the result to the MSEE assessor.

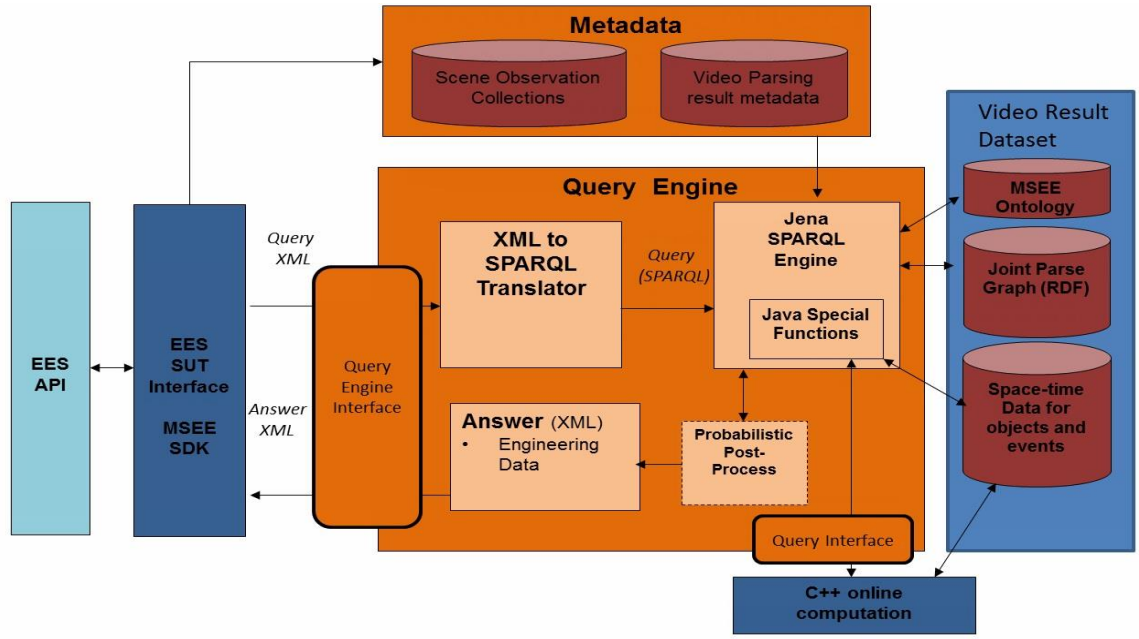



Figure 52. The architecture of the query answering engine for MSEE tests. The query engine (at the core) includes four components. The online C++ computation module extract certain relations asked by the query which are not pre-computed in the joint parse graph.

### 3.5.1. Data structures for the database

There are four main sets of data required by the Query Engine Module to answer queries:

- **Scene Observation Collections** (SOC) – this is a metadata of the sensor data, including sensor identifier, time-stamp, frame-rate, and calibration information.
- **MSEE Ontology** (MSEE.owl) – this is the ontology of general knowledge base describing the taxonomy of entities (objects, events), and relations between entities (e.g. supertype-subtype relations).
- **Joint Parse Graph** – this is the result of video parsing represented in RDF format.
- **Space-Time data for objects and events** – this consists of time-stamped world coordinate positions of the trajectories of objects and events.

#### 3.5.1.1. Parse graph represented as RDF

A parse graph can be represented as a Resource Description Framework (RDF) data model --- one of the standard knowledge representations in the semantic web. SPARQL is the standard language for querying RDF data. In the RDF data model, each statement has a triple format: *subject-predicate-object*. The collection of these statements intrinsically represents a directed graph. Therefore, the RDF is able to represent the video Joint Parse Graph naturally and allows the data to be queried via SPARQL.

#### 3.5.1.2. Space-time data

This dataset consists of time-stamped positions of objects and events in and scene, based on the result of object tracking and event analysis modules. Some of the queries require online processing of tracking data. An example of such a query is: "*did the car passes the person*?" Since such a query can be asked for any two objects in the scene, it can be prohibitively expensive to analysis every pair of objects to determine whether there is a passing event. Hence, an online computation is preferred, such that passing event inference is made only in response to a query. Similarly, a location-related query, such as "*is there a talking event in location-A*?" can only be answered online by examining the object/event positions with respect to the location specified in the query. Therefore, to answer these types of queries, the Query Engine needs to assess the object and event space-time data.

### 3.5.2. XML to SPARQL translator

To evaluate our system's ability to interpret the information obtained by the cameras, we provided with true/false queries in a restricted language defined in the *FLS* document. These queries are presented in an XML format specified in the *ICD* document. In order to obtain the answer to the query from the information that we have derived from the sensors, which is RDF data and tracking data, we convert the query into a format called SPARQL

The process of the XML to SPARQL Translator module, which translates the incoming XML query into a functionally equivalent SPARQL query, is as follows:

1) Generate the SPARQL “PREFIX” statements.

2) Translate the time definitions from XML to SPARQL.

3) Translate the location definitions from XML to SPARQL

4) Translate the set definitions from XML to SPARQL.

5) Translate the event definitions from XML to SPARQL.

6) Translate the XML query statement into SPARQL.

### 3.5.3. Jena SPARQL engine

Jena is an open-source Java framework for building Semantic Web applications, and ARQ is a query engine in Jena that supports the SPARQL query language. The translated queries in SPARQL forms are passed to the Jena ARQ library to answer the queries. The solution is retrieved based on graph pattern matching, by comparing the query triple conditions with the RDF data.

### 3.5.4. Query interface

The Query Engine Interface affords a standard means for accessing the Query Engine during evaluation. As Figure 53 shows, the evaluation process is defined (hierarchically) in terms of scene observation collections (SOCs), storylines and queries. Thus, the interface provides methods for starting a new SOC, for resetting state between storylines within a given SOC, and for making queries for a given storyline. Note that two query methods are provided, allowing queries and results to be stored either in memory or on disk. In either case, a given query is automatically translated from XML to SPARQL, prior to execution.

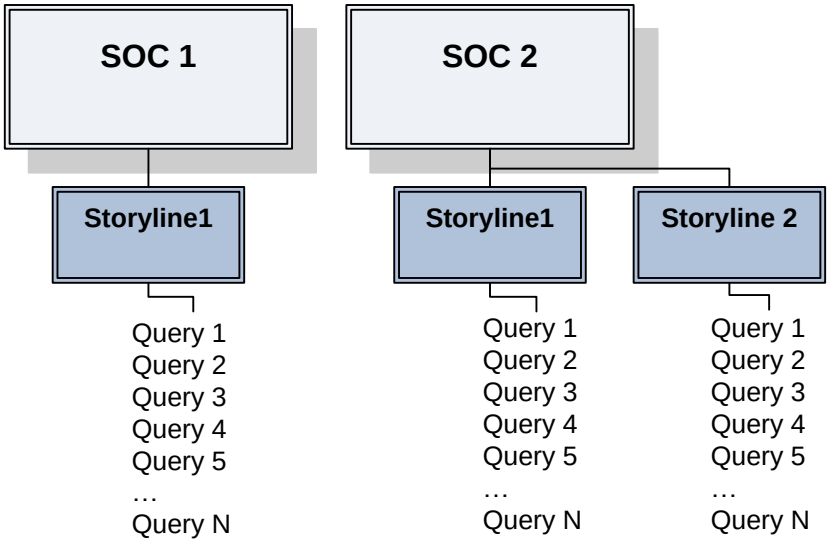



Figure 53. Structure of SOCs, storylines, and queries. This figure was obtained from the MSEE EES-SUT Interface Control Document (ICD), Version 1.1.

At present, the Query Engine Interface methods are invoked by one of two modules. In the integrated MSEE system, the methods are called by the MSEE SDK. However, they can also be called by a specialized Unit Tester module.

### 3.6 Evaluation Results and Analyses

#### 3.6.1 Overall Results

| Category | # Queries |
|---|---|
| Object definition | 243 |
| Classification | 71 |
| Part Of | 93 |
| Spatial | 58 |
| Attributes | 165 |
| Relationships | 291 |
| Tracking | 134 |
| Multiple predicate categories | 5 |

Table 12. Number of queries by category. The number of predicates used in each query can serve as a proxy for complexity of the query. Number of predicates is not a perfect measure of the complexity of a query because not all predicates are equally complex, and other factors affect query complexity (such as the size of the temporal and spatial windows that must be considered in answering the query).

The SUT is evaluated by a 3rd party company, SIG. We report the phase III evaluation results and analyses in this section. There are 1,160 polar queries as listed in Table 12. During the evaluation, our system did not utilize the ground-truth answers after answering each query for consecutive queries. Among the 1,160 queries, 243 queries are object definitions, 197 (81%) of which are successfully detected (which outperforms the state-of-the-art deep learning methods as shown in Table 2). For non-definition queries, we either provided binary ``true/false" answers or claimed ``unable to respond" (when our implementation cannot handle or recognize some of the predicates involved in a query). Table 13 shows the accuracy as the ratio of correctly answered queries to number of the responded non-definition queries.

| | Office | Parking lot (winter) | Parking lot (fall) | Garden | Auditorium |
|---|---|---|---|---|---|
| Video length | 17:35:36 | 8:14:42 | 4:27:44 | 4:15:56 | 8:53:24 |
| # of cameras | 12 | 12 | 11 | 8 | 11 |
| # moving cameras | 0 | 2 | 1 | 1 | 2 |
| # IR cameras | 0 | 1 | 1 | 0 | 1 |
| # of queries | 108 | 247 | 236 | 215 | 254 |
| Definition queries | - | 63 | 71 | 54 | 55 |
| Non-definition queries | 108 | 184 | 165 | 161 | 199 |
| Respond rate | 0.522 | 0.600 | 0.795 | 0.683 | 0.731 |
| Accuracy | 0.785 | 0.615 | 0.626 | 0.586 | 0.684 |

Table 13. Performance by data collection in Phase III evaluation by SIG.

Figure 54 further breakdowns the accuracy by the category of predicates and the number of unique predicates in a query. Most queries have either one, two, or three predicates. This is a natural result of the choice to avoid overcomplicating the queries. Queries with one predicate focus on various types of objects (people, car, etc.): most of these queries (243) are object definitions; the others (46) are about counting (e.g., ``how many people are in the scene?''). Queries with two predicates mostly involve attributes and properties of single objects: one predicate of the two is used to define the object (usually person or automobile), the other unary predicate focuses on attributes. Queries with three predicates focus on binary relationships operating on two objects: two predicates are used to define the operands and the third predicate is for relationships. The results reveal that our prototype system performs well in object detection tasks but requires more future work for correctly answering complex queries regarding spatial reasoning and interactions between entities.

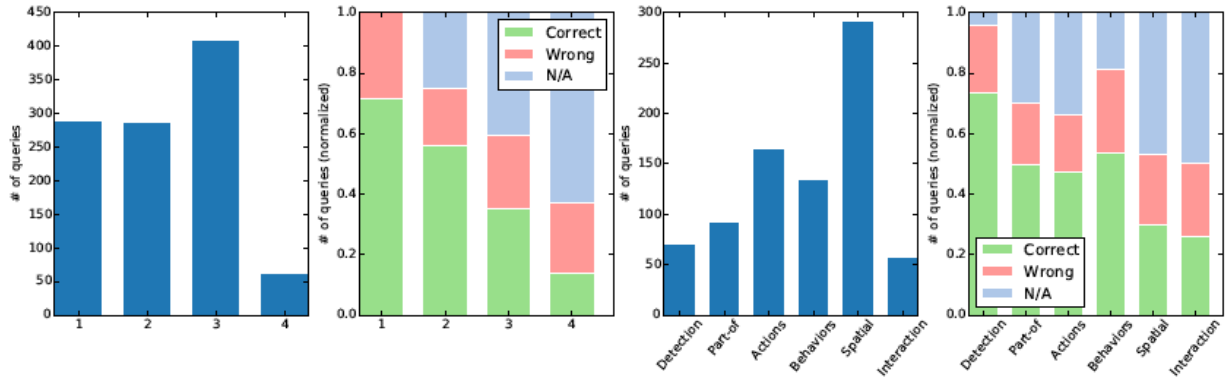



Figure 54. Results breakdown. Left : accuracies by the category of predicates. Right: accuracies by the number of unique predicates in a query.

Typical results are shown in the following figures for several SOCs.

- The 2D images on the top show the camera views and are augmented with bounding boxes for detected objects (Human, vehicles, animals etc) with attributes.
- The 3D scene visualizes the icons of humans, vehicles together with the actions, events, and 3D relations between them.
- For the mobile cameras, we estimate their Field-of-View (FoV) over time, and registered to the ground plane. Then we further estimate the position of the objects capture in the FoVs, and register these objects in the ground plane.


Figure 55. The parking lot scene with a moving camera (the upper left) mounted on a vehicle. 3D space and time parsing the scene and events by integrating images from multiple cameras.


Figure 56. The garden scene. 3D space and time parsing the scene and events by integrating images from multiple cameras.

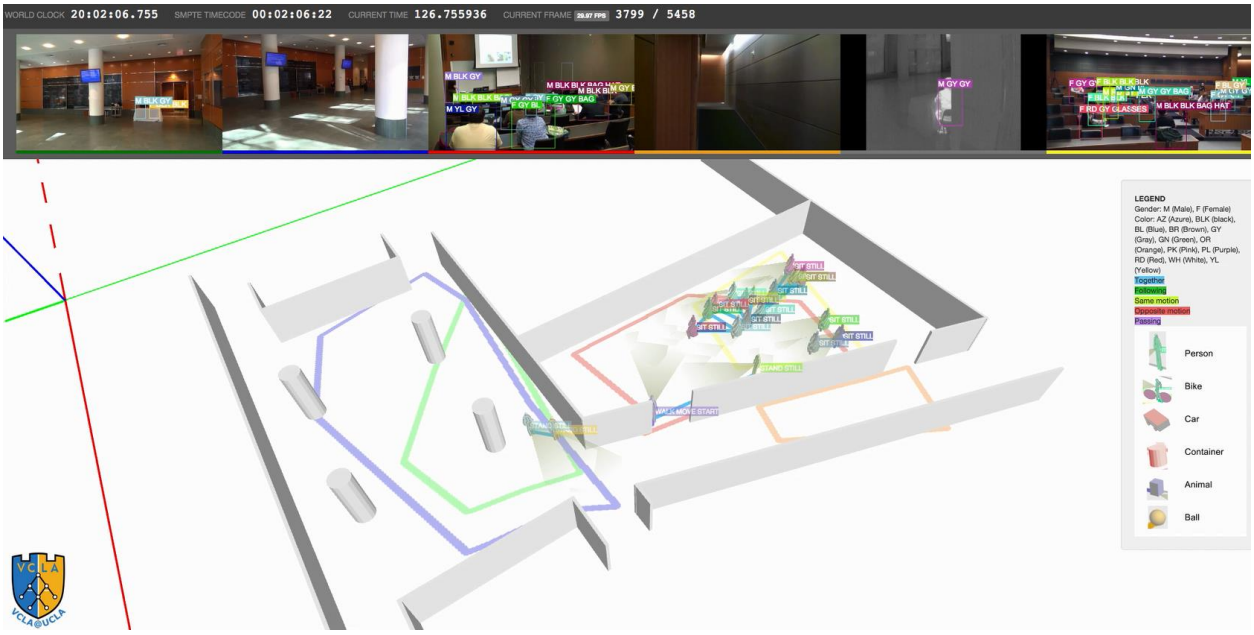



Figure 57. The auditorium scene. This includes the IR camera (the 5th on the top row). 3D space and time parsing the scene and events by integrating images from multiple cameras.

### 3.6.2 Evaluation Timeline

On April 3rd, 2015 the Phase 3.0 Testing Data stored on an external hard disk drive was shipped to UCLA, DARPA, and AFRL, with an expected arrival date no later than April 6th. UCLA was allotted two weeks to perform data preprocessing required by their SUT. Table 6 provides details on the preprocessing performed and the associated time durations, as reported by UCLA. BAE Systems exposed the EES interface for Phase 3 Evaluation on 12:01 AM EDT April 20, 2015. UCLA started its lone EES session at 5:30 PM EDT on April 21, and completed it at 8:55 PM EDT on April 28. After processing the queries associated with the first SOC, "soc-sig-office-2013-09-04-testing", UCLA paused the evaluation at 11:52 PM EDT on 4/22/15 to address interfacing and other SUT issues. Note that the "soc-sig-office-2013-09-04-testing" SOC is not part of the Phase3 testing data sets. UCLA resumed the evaluation at 3:27 AM EDT 4/26/15 and completed the evaluation at 8:55 AM EDT 4/28/15. More details are referred to the SIG reports.

Table 14 shows the log of processing time. Note that the run time was logged in phase III testing (HD videos 1920*1028 with intensive activities). Some modules used 80 CPU cores (including detection, tracking). The run time of detection is proportional to the size of a frame, and the run time of remaining modules (tracking, attributes and actions, etc.) is proportional to the number of total object instances. In our on-going work after phase III, we are refactoring our code. Some modules are re-implemented using GPUs. E.g., object detection module can run in real time using a NVIDIA GPU (Titan or K40).

| Reported Data Preprocessing Metric | SIG Office 2013-09-04 | SIG Parking Lot 2014-01-04 | SIG Parking Lot 2014-10-18 | Pratt Garden 2014-09-20 | Schiciano Auditorium 2014-02-22 |
|---|---|---|---|---|---|
| **Video duration** | 17:35:36 | 8:14:42 | 4:27:44 | 4:15:56 | 8:53:24 |
| **Total # of frames** | 2,486,289 | 888,053 | 481,414 | 458,629 | 959,216 |
| **Detection** | | | | | |
| **Human bounding boxes** | 1,341,704 | 1,885,106 | 487,808 | 2,718,738 | 2,433,349 |
| **Car bounding boxes** | N/A | 204,238 | 1,212,573 | N/A | N/A |
| **Bicycle bounding boxes** | N/A | 4,801 | 13,192 | 43,291 | N/A |
| **Processing time** | unreported | ~16 hours | ~19 hours | ~13 hours | ~15 hours |
| **Tracking** | | | | | |
| **Generated human tracks** | 2227 | 17,547 | 3,061 | 16,964 | 11,860 |
| **Generated car tracks** | N/A | 437 | 1,490 | N/A | N/A |
| **Generated bicycle tracks** | N/A | 92 | 186 | 321 | N/A |
| **Processing time** | unreported | ~34 hours | ~9 hours | ~33 hours | ~22 hours |
| **Attributes** | | | | | |
| **Processed human bounding boxes** | 1,340,395 | 1,236,245 | 486,879 | 2,270,448 | 2,429,963 |
| **Generated attribute boxes** | 4,057,955 | 4,442,954 | 1,574,064 | 8,194,540 | 7,667,580 |
| **Processing time** | ~18 hours | ~34 hours | ~6 hours | ~32 hours | ~22 hours |
| **Action** | | | | | |
| **Processed bounding boxes** | 1,330,240 | 1,884,974 | 487,614 | 2,718,600 | 2,433,209 |
| **Processing time** | ~20 hours | ~34 hours | ~6.5 hours | ~33 hours | ~23 hours |
| **Behavior** | | | | | |
| **Processed bounding boxes** | 1,217,572 | 1,993,022 | 1,664,075 | 2,594,942 | 2,246,625 |
| **Processing time** | ~13 min. | ~27 min. | ~17 min. | ~33 min. | ~33 min. |

Table 14. Summary of UCLA SUT Data Processing.

### 3.6.3 Performance Analyses

Table 15 shows the performance of the UCLA SUT broken down by SOC. “Object definition” queries are excluded from the metrics reported in this table. Note that the “SIG-Office 2013-09-04” SOC was used in the Phase 2.2 evaluation, though the queries presented in the Phase 3 evaluation are new. Note that object definition queries are not included in these results.

Table 16 summarizes performance metrics for sets of queries based on the number of predicates used in the query. Intuitively, we expect queries with more predicates to be more complex and therefore to have higher error rates. Once again, object definition queries are excluded from these results.

| Metric | SIG Parking Lot 2014-01-04 | SIG Parking Lot 2014-10-18 | Pratt Garden 2014-09-20 | Schiciano Auditorium 2014-02-22 | SIG Office 2013-09-04 |
|---|---|---|---|---|---|
| Number of queries | 184 | 165 | 161 | 199 | 108 |
| Number of responses | 96 (52.2%) | 99 (60.0%) | 128 (79.5%) | 136 (68.3%) | 79 (73.1%) |
| Confidence >= 0.0 | | | | | |
| Number of declarations | 96 | 99 | 128 | 136 | 79 |
| Declaration rate | 1 | 1 | 1 | 1 | 1 |
| Error rate | 0.385 | 0.374 | 0.414 | 0.316 | 0.215 |
| Confidence error | 0.121 | 0.076 | 0.051 | 0.112 | 0.087 |
| Brier score | 0.332 | 0.340 | 0.307 | 0.320 | 0.211 |
| Confidence >= 0.6 | | | | | |
| Number of declarations | 67 | 81 | 114 | 99 | 59 |
| Declaration rate | 0.698 | 0.818 | 0.891 | 0.728 | 0.747 |
| Error rate | 0.433 | 0.395 | 0.421 | 0.364 | 0.203 |
| Confidence error | 0.016 | 0.009 | 0.026 | 0.009 | 0.008 |
| Brier score | 0.343 | 0.343 | 0.314 | 0.312 | 0.183 |
| Confidence >= 0.9 | | | | | |
| Number of declarations | 28 | 63 | 24 | 96 | 50 |
| Declaration rate | 0.292 | 0.636 | 0.188 | 0.706 | 0.633 |
| Error rate | 0.429 | 0.381 | 0.333 | 0.365 | 0.200 |
| Confidence error | 0.010 | 0.004 | 0.002 | 0.006 | 0.003 |
| Brier score | 0.362 | 0.340 | 0.304 | 0.315 | 0.185 |

Table 15. Performance metrics by SOC.

| Metric | N=1 | N = 2 | N = 3 | N = 4 | 5 <= N < 10 |
|---|---|---|---|---|---|
| Number of queries | 46 | 287 | 410 | 62 | 12 |
| Number of responses | 46 | 215 | 243 | 23 | 11 |
| Confidence >= 0.0 | | | | | |
| Number of declarations | 46 | 215 | 243 | 23 | 11 |
| Declaration rate | 1 | 1 | 1 | 1 | 1 |
| Error rate | 0.283 | 0.284 | 0.395 | 0.478 | 0.545 |
| Confidence error | 0.099 | 0.116 | 0.066 | 0.043 | 0.094 |
| Brier score | 0.205 | 0.284 | 0.335 | 0.338 | 0.482 |
| Confidence >= 0.6 | | | | | |
| Number of declarations | 34 | 152 | 205 | 21 | 8 |
| Declaration rate | 0.739 | 0.707 | 0.844 | 0.913 | 0.727 |
| Error rate | 0.235 | 0.322 | 0.42 | 0.429 | 0.625 |
| Confidence error | 0.01 | 0.014 | 0.016 | 0.014 | 0.008 |
| Brier score | 0.183 | 0.261 | 0.344 | 0.355 | 0.543 |
| Confidence >= 0.9 | | | | | |
| Number of declarations | 27 | 93 | 122 | 13 | 6 |
| Declaration rate | 0.587 | 0.433 | 0.502 | 0.565 | 0.545 |
| Error rate | 0.185 | 0.28 | 0.393 | 0.462 | 0.667 |
| Confidence error | 0.003 | 0.005 | 0.006 | 0.005 | 0.003 |
| Brier score | 0.167 | 0.243 | 0.347 | 0.41 | 0.595 |

Table 16. Performance by number of predicates.

### 3.6.4. Summary of the failing conditions

In general, the major failure cases in Phase III test can be divided into five reasons.

i. **System coding and predicate definition issues.** There were some bugs during the online computing process.

ii. **3D cross camera fusion.** This module is newly added for Phase III, and was not accurate enough. This caused a certain portion of errors.

iii. **3D pose and relations**. The estimation of 3D human pose is not accurate to answer questions in 3D, such as a person touching another person or object. The 3D relations in indoor scenes are also unreliable due to heavy occlusions by furniture.

iv. **Recognition in very low resolution**. This causes a large portion of the mistakes in answering queries. Some cases can be resolved by long-range spatial and temporal contextual information or model. For example, when you detect a person is probably drinking in one time, and then we can infer his hand must hold a cup, bottle or a soda can.

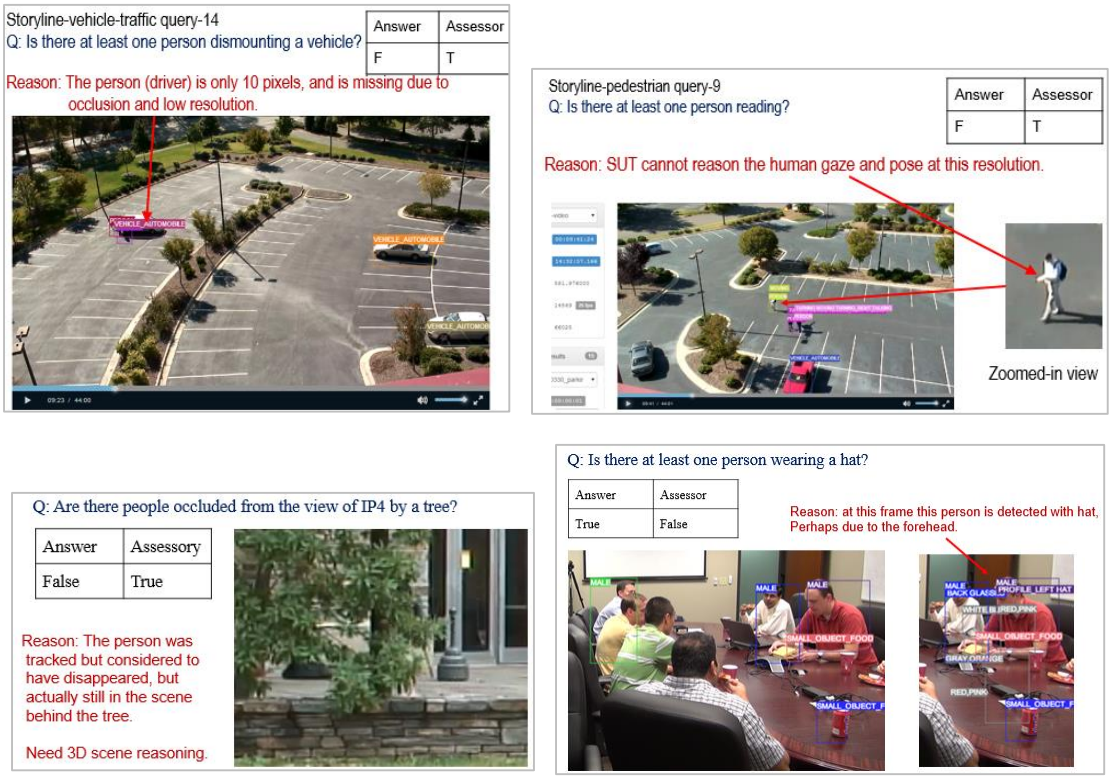



Figure 58. Some examples of failure.

Figure 58 shows some examples of failure. We address some of the issues after Phase III test.

# 4. New Developments after Phase III Test

After Phase III evaluation, we continued to improve the system under test (SUT). We developed a new web-based query-answering interface which substitutes the stand-alone module developed by SIG in the evaluation. We improved the performance of several vision muddles in the parsing pipeline. We developed a new graph knowledge database for storing parsing results and more efficient query answering.

## 4.1. A Web-based Query-Answering Interface

The interface is shown in Figure 54 to Figure 56. On the top, we show the view-based parsing results (4 views in the garden scene in the figure), which include results from object detection, human, pose parsing and human attribute recognition, action and event detection and parsing, etc. On the bottom, we have several functional tabs:

- Object identification interface, which allows users to initialize a storyline query by defining an object in one view through a 2D bounding box or a 2D point.

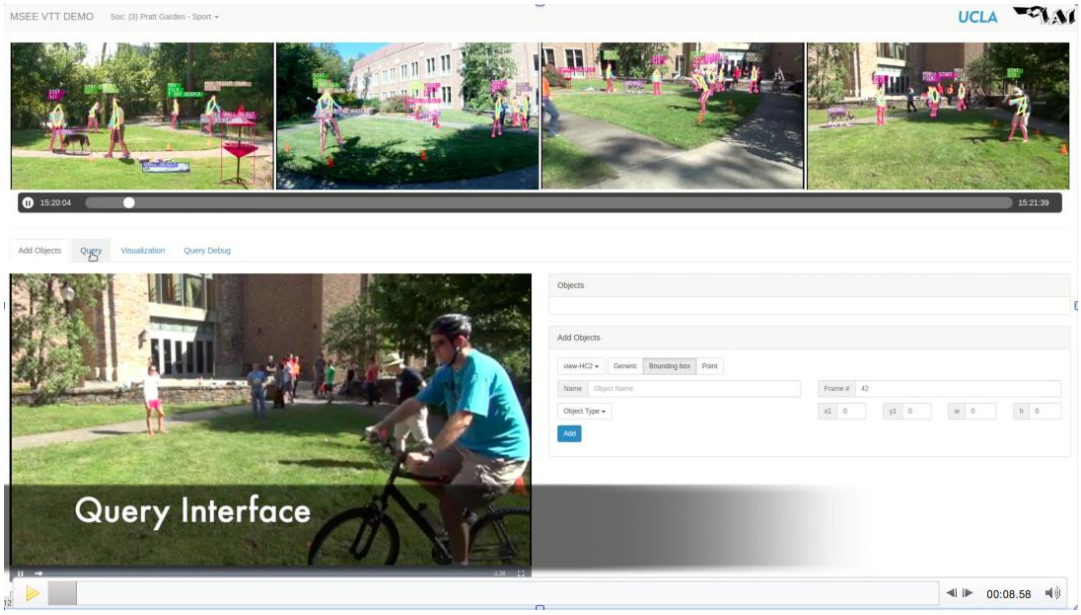



Figure 54. Object identification interface.

- Query interface, which allows users to compose ontology-guided queries quickly. Queries can attribute-based predicates or multi-object relationship based composite queries in single view or multiple views and/or across time.

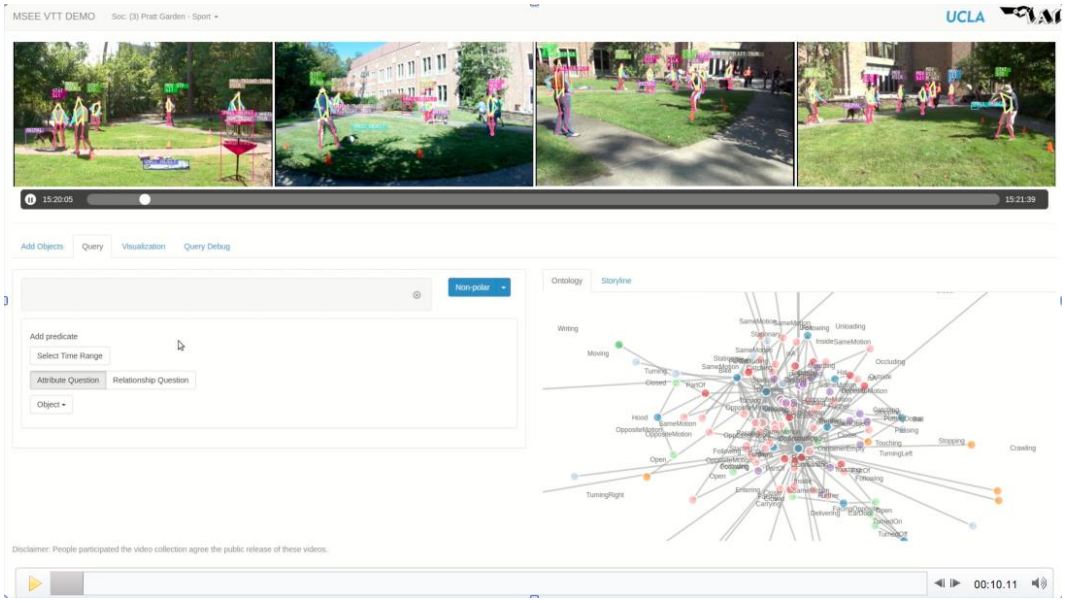



Figure 55. Ontology-guided query-composing interface.

- Visualization interface, which shows 3D scene based results (on the left) and query results (on the right), as shown in the bottom figure. (iv) Query debugging interface, which shows all the intermediate results during the query-answering process.

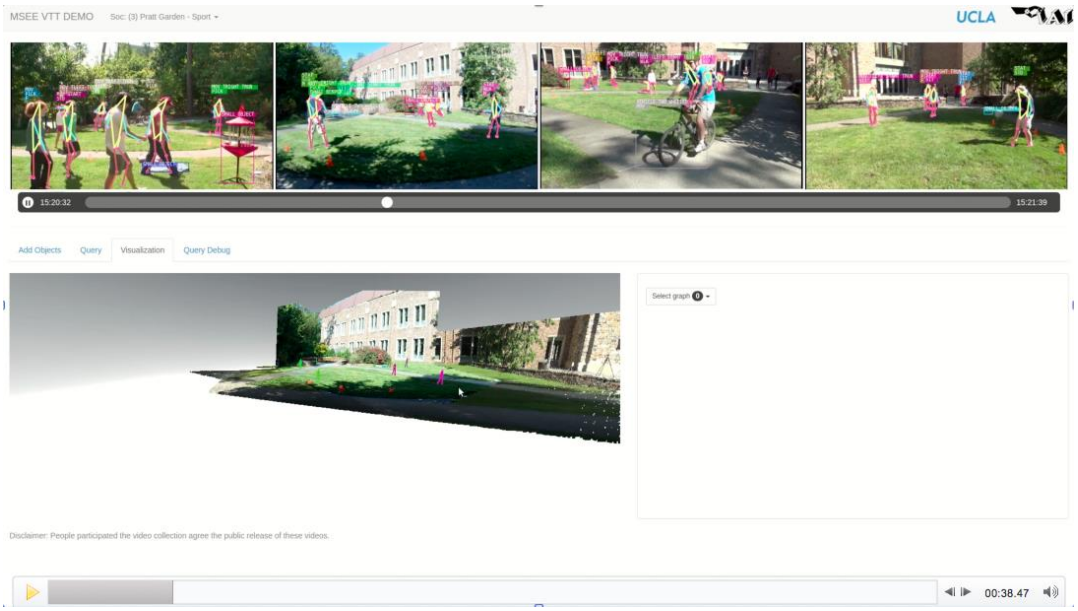



Figure 56. Visualization interface.

We developed an efficient method of composing queries with the guidance from the MSEE ontology. First, we represent the MSEE ontology by a graph as illustrated in Fig 57.

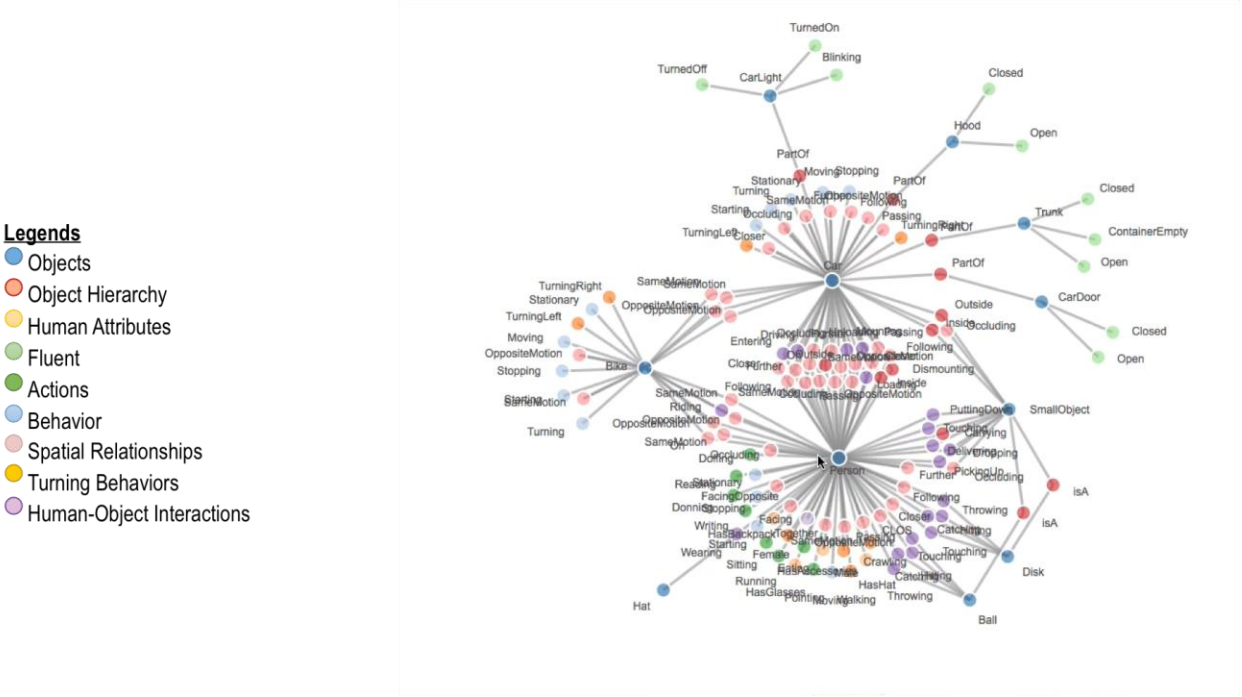



Figure 57. The ontology graph. For clarity, only a partition of the full graph is shown.

The ontology graph encodes what we can ask in the visual Turing test, which can be learned from the ConceptNet and different generic knowledgebase such as the Freebase, as well as domain-specific expert knowledgebase. Fig. 58 shows examples of automatically generating queries through random walk in the ontology graph. Fig. 59 shows an example of mapping queries created by users into the ontology graph so that a computer can understand what a user talk about.


Is there a female person (Mary)?

Is Mary catching a ball?

**Which** bike is turning right?

**Which** car which has the same motion with the bike?

Does Mary has a clear line of sight to the car?

Figure 58. Illustration of automatic query generation as random walk in the ontology graph.

**Which** car has a open door?

**Who** was wearing glasses and entered the car?

Was the person carrying a small object?

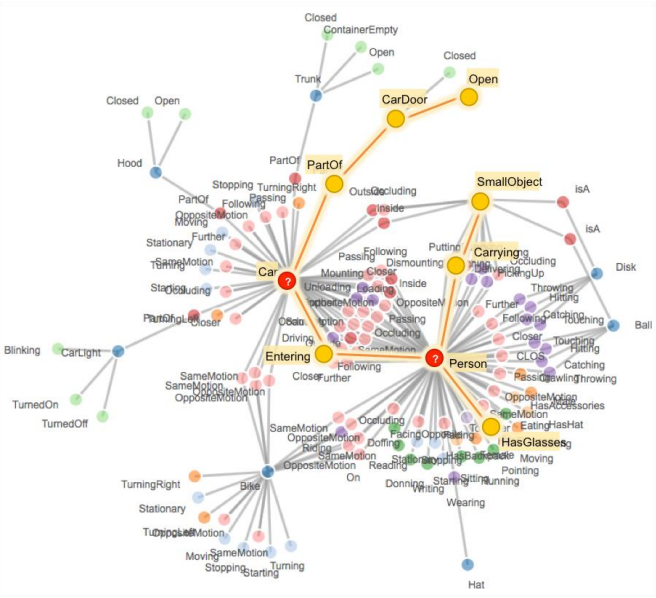



Figure 59. Illustration of mapping user created queries (left) into the ontology graph.

## 4.2. Improved Vision Modules

### 4.2.1. Multi-view Multi-object Tracking with 3D cues

We developed a hierarchical composition approach for multi-view object tracking. As illustrated in Fig. 60, the key idea is to adaptively exploit multiple cues in both 2D and 3D, e.g., ground occupancy consistency, appearance similarity, motion coherence etc., which are mutually complementary while tracking the humans of interests over time. While feature online selection has been extensively studied in the past literature, it remains unclear how to effectively schedule these cues for the tracking purpose especially when encountering various challenges, e.g. occlusions, conjunctions, and appearance variations. To do so, we propose a hierarchical composition model and re-formulate multi-view multi-object tracking as a problem of compositional structure optimization. We setup a set of composition criteria, each of which corresponds to one particular cue. The hierarchical composition process is pursued by exploiting different criteria, which impose constraints between a graph node and its offsprings in the hierarchy. We learn the composition criteria using MLE on annotated data and efficiently construct the hierarchical graph by an iterative greedy pursuit algorithm.

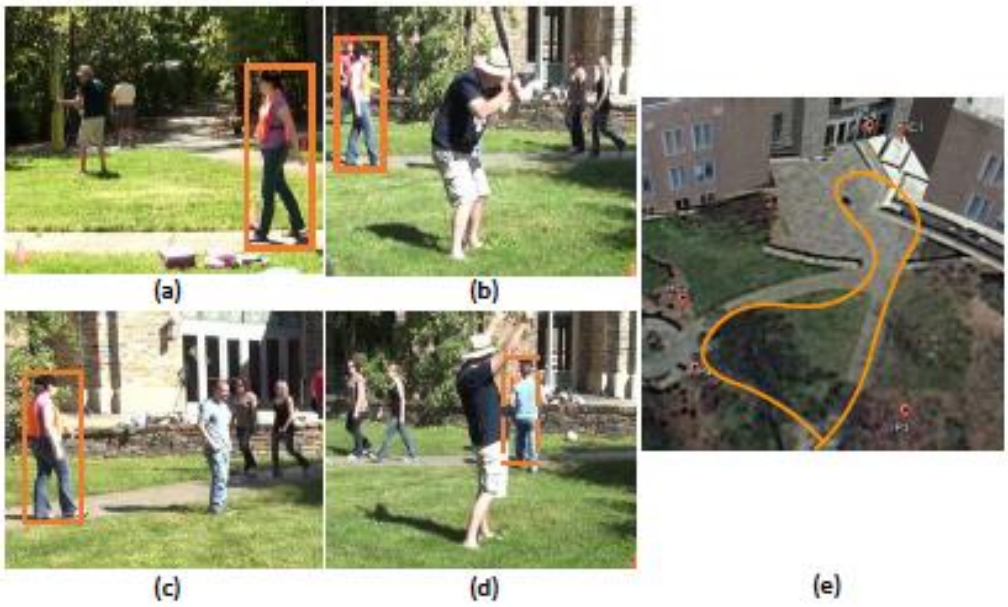



Figure 60. An illustration of utilizing different cues at different periods for the task multi-view multi-object tracking. suppose we would like to track the highlighted subject and obtain its complete trajectory (e). The optimal strategy for tracking may vary over space and time. For example, in (a), since the subject shares the same appearance within certain time period, we apply an appearance based tracker to get a 2D tracklet; in (b) and (c), since the subject can be fully observed from two different views, we can group these two boxes into a 3D tracklet by testing the proximity of their 3D locations; in (d), since the subject is fully occluded in this view, we consider sampling its position from the 3D trajectory curve constrained by background occupancy.

Fig. 61 shows the proposed hierarchical compositional structure. Figure 62 shows the ideas of finding feasible regions (polygons) for interacting people.

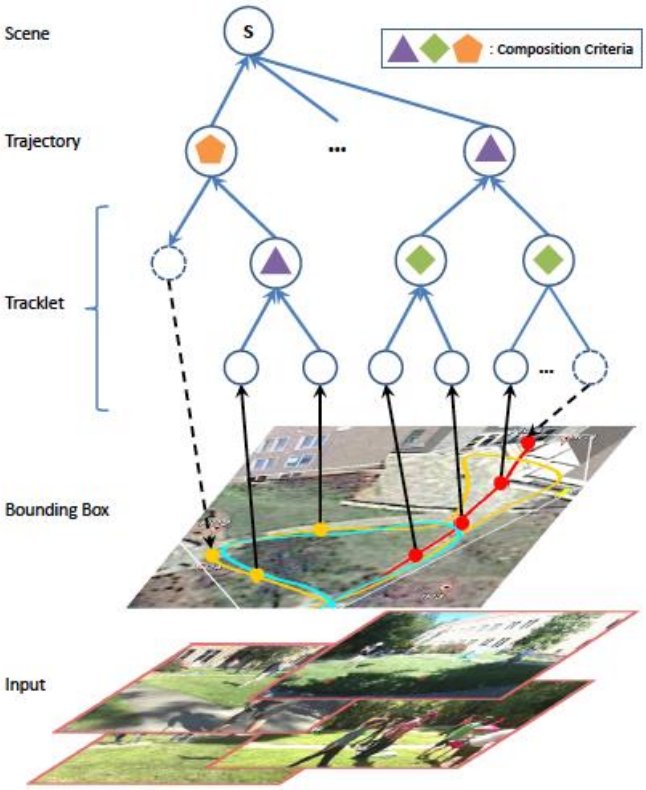



Figure 61. An illustration of the hierarchical compositional structure.

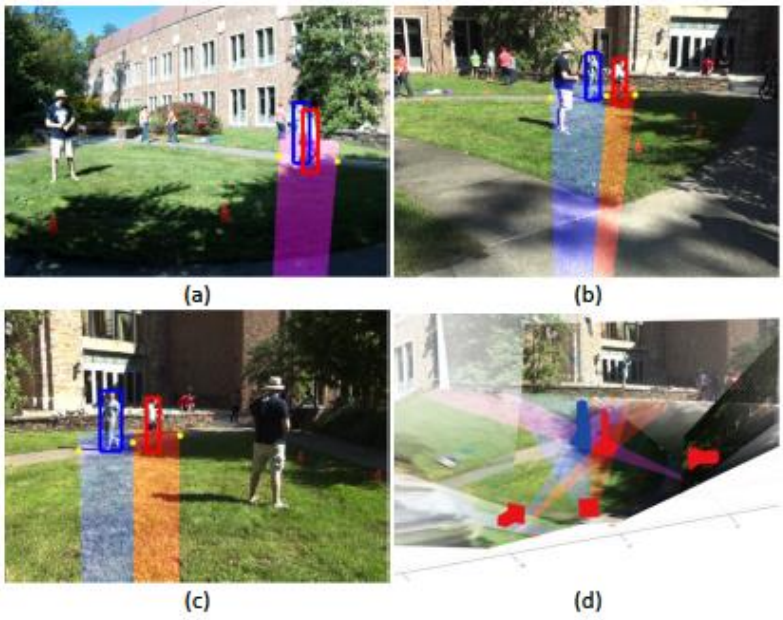



Figure 62. An illustration of finding feasible regions (polygons) for interacting people.

### 4.2.2. Joint Inference of Human Attributes and Poses

We developed a model for joint inferences of pose and attribute. The proposed algorithm has two properties.

- Explicitly representing the decomposition and articulation of body parts, and account for the correlations between poses and attributes in a common framework (A-AOG);
- Providing robust system to handle data with large variation of appearance and geometric, heave occlusion, and truncation.

  To achieve the first objective above, we design a unified framework to infer attribute and pose simultaneously. Most existing methods train models separately for each of these two tasks attribute classification and pose estimation, and combine the inference sequentially, e.g. first do pose estimation, then use the detected part locations to recognize the attribute. The main problem is that the attribute recognition deteriorates if the part locations are not detected correctly, so most previous methods need the ground-truth bounding box of the target person during testing. The inference process of our approach is illustrated in Figure 63.

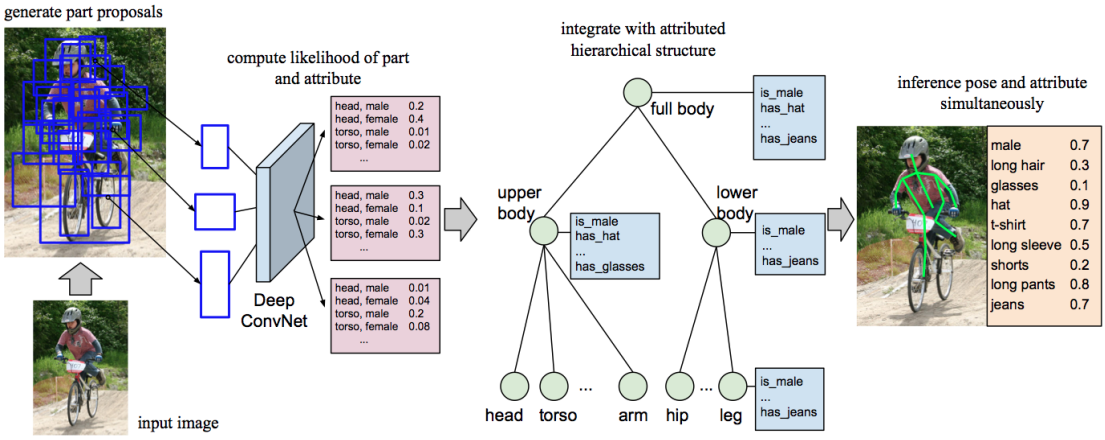



Figure 63. illustration of the unified inference of attribute and pose

To infer attribute and pose from an input image, we (1) first generate the part proposals by using a deep part proposal network; (2) compute the likelihood of each part-attribute combination at each proposal region using the deep part-attribute network; (3) integrate the part and attribute into an attributed hierarchical structure; and (4) infer pose and attribute efficiently and simultaneously by using dynamic programming.

To achieve the second object above, we propose a new way to design and propose parts. In previous pose-estimation approaches, people annotate the parts or draw the bounding box with same scale and aspect ratio, however, these approaches showed limitations when data has large variations of appearance and geometric, heavy occlusion, and truncation. To overcome this, we design the way to design parts according to the proposals and joint annotations.

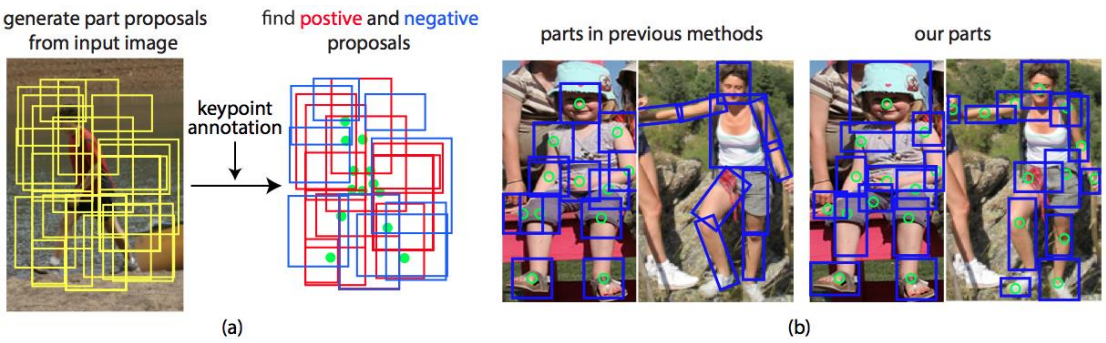



Figure 64. (a) generation of part proposals. (b) part design in previous methods and our method.

As shown in Figure 64. (a), we first generate part proposals from input image, and define the part labels of proposals by computing the distance between annotated joints and centers of proposals. We compare parts in our approach and and previous approaches in Figure 64. (b). Our parts have large number of different scales and aspect ratios compared to other methods thus we are more robust to large pose variation and heavy occlusion.

To evaluate our model, we test our model on popular attribute classification benchmark Poselet Attributes of People dataset, and we achieve state-of-the-art performance.

| Method | Male | Long hair | Glasses | Hat | T-shirt | Long sleeve | Shorts | Jeans | Long pants | mAP |
|---|---|---|---|---|---|---|---|---|---|---|
| Joo et al. [31] | 88.0 | 80.1 | 56.0 | 75.4 | 53.5 | 75.2 | 47.6 | 69.3 | 91.1 | 70.7 |
| PANDA [3] | 91.7 | 82.7 | 70.0 | 74.2 | 68.8 | 86.0 | 79.1 | 81.0 | 96.4 | 78.98 |
| Park et al. [11] | 92.1 | 85.2 | 69.4 | 76.2 | 69.1 | 84.4 | 68.2 | 82.4 | 94.9 | 80.20 |
| Gkioxari et al. (8.L) [4] | 91.7 | **86.3** | 72.5 | 89.9 | **69.0** | 90.1 | 88.5 | **88.3** | **98.1** | 86.0 |
| Ours w pose (8.L) | **91.9** | 85.0 | **79.7** | **90.4** | 65.5 | **92.1** | **89.9** | 87.3 | 97.9 | **86.7** |
| Gkioxari et al. (16.L) [4] | 92.9 | 90.1 | 77.7 | 93.6 | 72.6 | 93.2 | 93.9 | 92.1 | **98.8** | 89.5 |
| R* CNN (16.L) [21] | 92.8 | 88.9 | 82.4 | 92.2 | **74.8** | 91.2 | 92.9 | 89.4 | 97.9 | 89.2 |
| Ours w pose (16.L) | **94.9** | **90.6** | **85.2** | **93.7** | 71.3 | **95.1** | **94.2** | **93.1** | **98.8** | **90.7** |
| Gkioxari et al. (8.L) [4] | 84.1 | 77.9 | 62.7 | 84.5 | **66.8** | 84.7 | 80.7 | 79.2 | 91.9 | 79.2 |
| Ours w/o pose (8.L) | **88.3** | **84.1** | 73.2 | 86.4 | 57.1 | 90.1 | 78.8 | 85.1 | 95.8 | 81.6 |
| Ours w pose (8.L) | 87.9 | 83.6 | **75.4** | **87.3** | 62.2 | **92.1** | **84.1** | **87.6** | **97.6** | **84.2** |
| Gkioxari et al. (16.L) [4] | 90.1 | 85.2 | 70.2 | 89.8 | 63.2 | 89.7 | 83.4 | 84.8 | 96.3 | 83.6 |
| Ours w/o pose (16.L) | 92.1 | 88.4 | 76.4 | 90.1 | 62.7 | 92.8 | 82.5 | 89.2 | 98.1 | 85.8 |
| Ours w pose (16.L) | **93.7** | **91.1** | **78.5** | **92.6** | **68.2** | **94.0** | **88.4** | **92.1** | **98.6** | **88.5** |

Table 12. Comparison of attribute classification on Attributes of People dataset. 8.L and 16.L indicate 8 layer and 16 layer CNN model respectively.

All the methods above the double horizontal line use the ground-truth bounding box of the target person at test time, and the methods below are tested without ground-truth bounding box. We can achieve the best mean average precision among the other methods with the same number of layers of deep network, also the joint model (Ours w pose) can improve the model without pose (Ours w/o pose) around 3 percent which demonstrates the strength of our joint modeling of the two tasks.

### 4.3. A New Graph Database for Knowledge Representation

We adopted as backend a new state-of-the-art graph database – Neo4j mainly due to

- Multiple modules perform graph operations e.g. in video/text joint parsing, dialogue management, planning
- Many graph operations are common across modules, e.g.
- Create or modify graph nodes and edges
- Search data based on graph patterns

With Neo4j, we aim at the following three objectives:

- Standardize graph data representation for all modules, e.g. representation of object, event/task, attribute and relation
- Provide a single graph database as a shared workspace for all modules to store and exchange information
- Provide a common API for efficient high-level graph operations

Neo4j graph database has several desirable properties:

- Highly scalable open source graph database
- Most popular graph database in-use today
- Query using Cypher query language which is intuitive for graph pattern matching
- API support for Java and Python

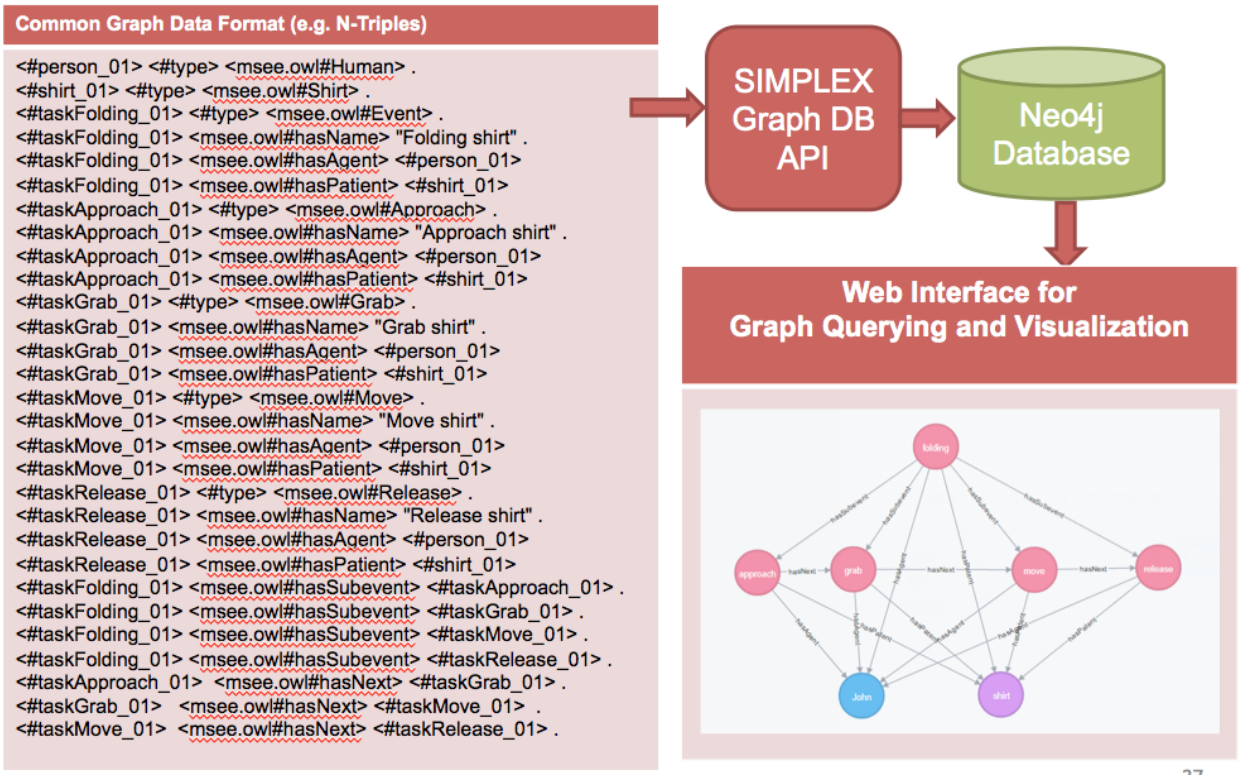



Figure 65. An example of data importing to Neo4j graph database.

We used Neo4j database as the software representation layer for knowledge representation and robot learning. We developed a Java-based "SIMPLEX GraphDB API" for high-level graph operations. We also developed common querying and visualization user interfaces. Figure 65 shows an example of data importing to Neo4j. The graph operations of interests in this project include:

- **Data manipulation**
- Create, modify, delete graph node, edges and attributes
- **Search and Query**
- Search for data using graph patterns
- Spatial and temporal reasoning to answer queries
- **Co-reference**
- Matching of entities across different domains, e.g. text, video
- **Merging**
- Merging multiple sub-graphs to form a joint graph, e.g.
  - Joint video-text parse graph
  - A temporal And-Or graph from multiple video parsed graph

- Important for generalization
- **Analysis**
- Checking for consistency
- Concept and rule discovery

As illustrated in Figure 66, the graph operations can be implemented by the Cypher graph query language provided by Neo4j, which is a declarative and SQL-inspired language for describing patterns in graphs and uses syntax that looks like graph pattern.

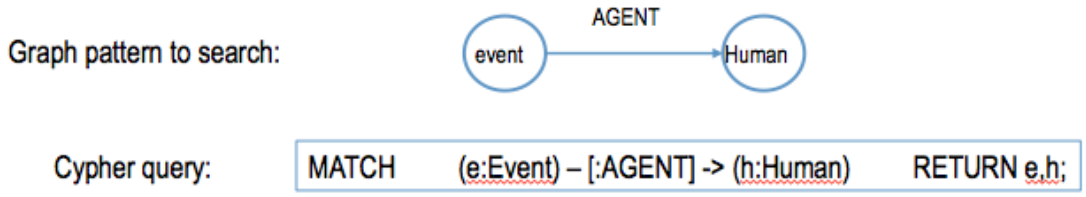



Figure 66. Illustration of the Cypher graph query language in Neo4j.

To leverage the parsing pipeline demonstrated in MSEE, we developed a VEML-to-Cypher conversion module for parsing natural language questions to Cypher query, as is illustrated in Figure 67.

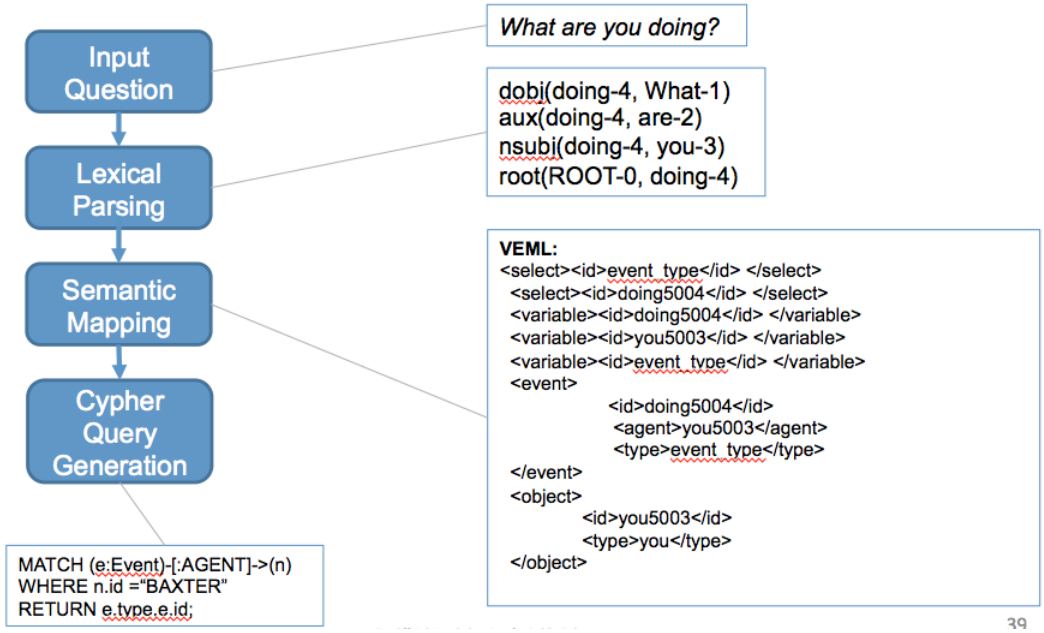



Figure 67. The VEML-to-Cypher conversion.

Figure 68 shows an example of querying the Neo4j graph database.

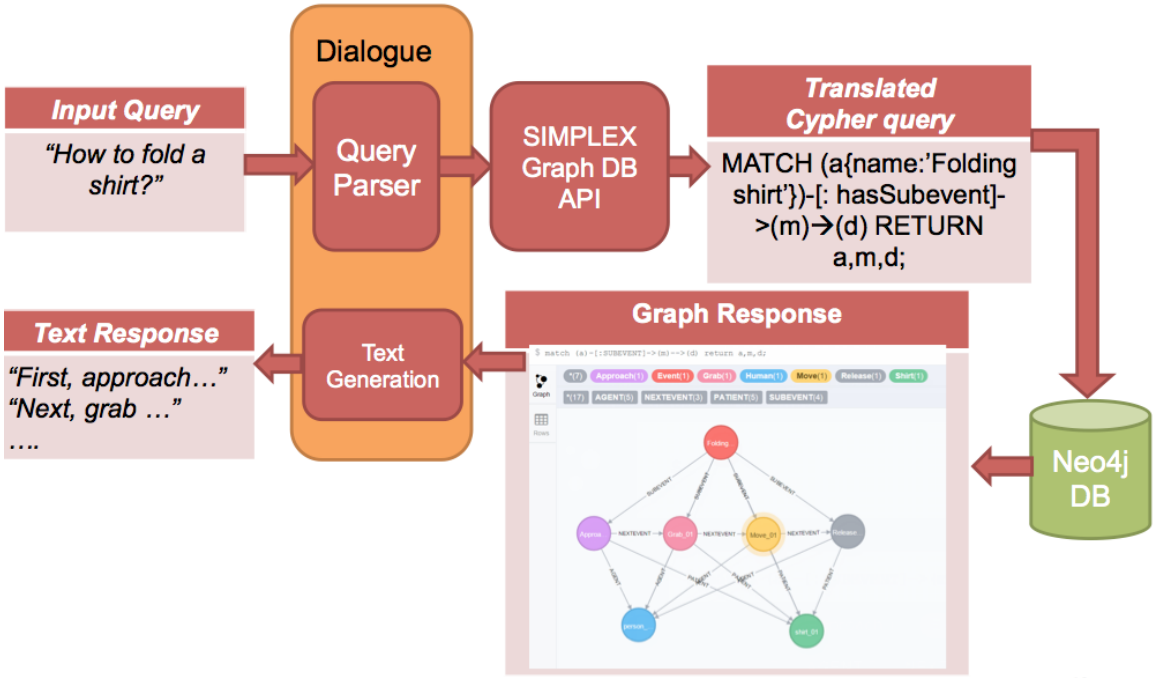



Figure 68. An example of querying Neo4j graph database.

## 5. Publications generated from this project effort


[1]. H.F. Gong and S.C. Zhu, "Intrackability: Characterizing Video Statistics and Pursuing Video Representations," Int'l Journal of Computer Vision, vol. 97, no. 3, 255-275, 2012.

[2]. Z. Si and S.C. Zhu, "Learning Hybrid image Template (HiT) by Information Projection," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 34, no.7, 1354-1367, 2012.

[3]. N. Jiang, W. Liu and Y. Wu, "Order Determination and Sparsity-Regularized Metric Learning for Adaptive Visual Tracking", in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) , 2012.

[4]. W. Z, Hu, "Learning 3D Object Templates by Hierarchical Quantization of Geometry and Appearance Spaces," in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) , 2012.

[5]. J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Mining Actionlet Ensemble for Action Recognition with Depth Cameras," in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) , 2012.

[6]. M. R. Amer and S. Todorovic, "Sum-Product Networks for Modeling Activities with Stochastic Structure," in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) , 2012.

[7]. A. Shrivastava and P. Li, "Fast Near Neighbor Search in High-Dimensional Binary Data," in European Conf. on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, 2012.

[8]. M. Amer, D. Xie, M. Zhao, S. Todorovic and S.C. Zhu, “Detecting and Localizing Activities at Different Scales under Budget,” in European Conference on Computer Vision (ECCV), 2012.
[9]. Y. B. Zhao and S.C. Zhu, “Scene Parsing by Integrating Function, Geometry and Appearance Models,” in Proceedings of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June, 2013.
[10]. B. Rothrock, S. Park, and S.C. Zhu, “Integrating Grammar and Segmentation for Human Pose Estimation,” in Proceedings of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June, 2013.
[11]. X. Song, T.F. Wu, Y. Jia, and S.C. Zhu, “Discriminatively Trained And-Or Tree Models for Object Detection,” in Proceedings of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June, 2013.
[12]. S. Wang, J. Joo, Y.Z. Wang, and S.C. Zhu, “Weakly Supervised Learning for Attribute Localization in Outdoor Scenes,” Proceedings of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June, 2013.
[13]. A. Fire and S.C. Zhu, “Learning Perceptual Causality from Video,” in AAAI Workshop on Learning Rich Representations from Low-Level Sensors (RepLearning), Bellvue, WA, July, 2013.
[14]. Z. Chen, H. Jin, Z. Lin, S. Cohen and Y. Wu, "Large Displacement Optical Flow from Nearest Neighbor Fields", in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June, 2013.
[15]. X. Shen, Z. Lin, J. Brandt and Y. Wu, "Detecting and Aligning Faces by Image Retrieval", in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June, 2013.
[16]. A. Fire and S.C. Zhu, “Using Causal Induction in Humans to Learn and Infer Causality from Video,” In 35th Annual Cognitive Science Conference (CogSci), Berlin, Germany, August, 2013.
[17]. A. Barbu, M. Pavlovskaia and S.C. Zhu, “Rates for Inductive Learning of Compositional Models,” In AAAI Workshop on Learning Rich Representations from Low-Level Sensors (RepLearning), Bellvue, WA, July, 2013.
[18]. B. Yao, B. Nie, Z. Liu, and S.C. Zhu. “Animated Pose Templates for Modeling and Detecting Human Actions,” IEEE Trans on Pattern Analysis and Machine Intelligence, 2013.
[19]. K.W Tu, M. Pavlovskaia and S.C. Zhu. “Unsupervised Structure Learning of Stochastic And-Or Grammars,” NIPS 2013.
[20]. D. Xie, S. Todorovic and S.C. Zhu. “Inferring ‘Dark Matter’ and ‘Dark Energy’ from Videos,” ICCV 2013.
[21]. T.F. Wu and S.C. Zhu. “Learning Near-Optimal Cost-Sensitive Decision Policy for Object Detection,” ICCV 2013.
[22]. M. Amer, S. Todorovic, A. Fern and S.C. Zhu. “Monte Carlo Tree Search for Scheduling Activity Recognition,” ICCV 2013.
[23]. J. Joo, S. Wang and S.C. Zhu. “Human Attribute Recognition by Rich Appearance Dictionary,” ICCV 2013.
[24]. J. Dai, Y. Wu, J. Zhou and S.C. Zhu. “Cosegmentation and Cosketch by Unsupervised Learning,” ICCV 2013.

[25]. P. Wei, N.N. Zheng, Y.B. Zhao and S.C. Zhu. “Concurrent Action Detection with Structural Prediction,” ICCV 2013.
[26]. B. Li, W. Hu, T.F. Wu and S.C. Zhu. “Modeling Occlusion by Discriminative AND-OR Structures,” ICCV 2013
[27]. P. Wei, Y. B. Zhao, N.N. Zheng and S.C. Zhu. “Modeling 4D Human-Object Interactions for Event and Object Recognition,” ICCV 2013.
[28]. T.E. Choe, H. Deng, F. Guo, M. Lee, and N. Haering, "Semantic Video-to-Video Search Using Sub-graph Grouping and Matching", 5th International Workshop on Video Event Categorization, Tagging, and Retrieval (VECTaR 2013), held in conjunction with ICCV, 2013.
[29]. Y. Hong, Z.Z. Si, W.Z Hu, S.C. Zhu and Y.N. Wu, “Unsupervised Learning of Compositional Sparse Code for Natural Image Representation,” Quarterly of Applied Mathematics, published online in November, 2013.
[30]. M.T. Pei, Z.Z. Si, B. Yao, and S.C. Zhu, “Video Event Parsing and Learning with Goal and Intent Prediction ” Computer Vision and Image Understanding, vol. 117, no. 10, pp 1369-1383, 2013.
[31]. Z. Si and S.C. Zhu, “Learning And-Or Templates for Object Modeling and Recognition,” IEEE Trans on Pattern Analysis and Machine Intelligence, vol. 35, no.9, 2189-2205, 2013.
[32]. X. Liu, Y. Zhao, S.C. Zhu, “Single-view 3D Scene Parsing by Attributed Grammar,” IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June, 2014.
[33]. K. Tu, M. Meng, M. W. Lee, T.E. Choi, and S.C. Zhu, “Joint Video and Text Parsing for Understanding Events and Answering Queries,” IEEE Multimedia, vol.21, no 2, pp42-70, may, 2014.
[34]. B. Yao, B. Nie, Z. Liu, and S.C. Zhu, “Animated Pose Templates for Modeling and Detecting Human Actions,” IEEE Trans on Pattern Analysis and Machine Intelligence, Vol. 36, No 3, March, 2014.
[35]. J. Wang, B. Nie, Y. Wu and S.C. Zhu, “Cross-view Action Modeling, Learning and Recognition,” IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June, 2014.
[36]. Y. Lu, T.F. Wu, and S.C. Zhu, “Online Object Tracking, Learning and Parsing with And-Or Graphs,” IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June, 2014.
[37]. J. Xie. W. Hu, S.C. Zhu and Y. Wu, “Learning Inhomogeneous FRAME Models for Object Patterns,” IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June, 2014.
[38]. J. Dai, Y. Hong, W. Hu, S.C. Zhu and Y. Wu, “Unsupervised Learning of Dictionaries of Hierarchical Compositional Models,” IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June, 2014.
[39]. B. Zheng, Y.B. Zhao, J. C. Yu, K. Ikeuchi and S.C. Zhu, “Detecting Potential Falling Objects by Inferring Human Action and Natural Disturbance,” Proc. Int'l Conf. on Robotics and Automations (ICRA), June, 2014.
[40]. B. Li, T.F. Wu and S.C. Zhu, Integrating Context and Occlusion for Car Detection by Hierarchical And-Or Model, Proc. of European Conf. on Computer Vision (ECCV), Dec. 2014.
[41]. T.F. Wu and S.C. Zhu, Learning Near-Optimal Cost-Sensitive Decision Policy for Object Detection, IEEE Trans. on PAMI, vol. 37, no. 5, pp 1013-1027, 2015.

[42]. M. Pavlovskaia, K.W. Tu and S.C. Zhu, Mapping Energy Landscapes of Non-Convex Learning Problems, Proc. of Energy Minimization Method for Computer Vision and Pattern Recognition (EMMCVPR), Hong Kong, Jan. 2015.
[43]. B. X. Nie, C. Xiong and S.C. Zhu, Joint Action Recognition and Pose Estimation From Video, Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Boston, June 2015.
[44]. T. Shu, D. Xie, B. Rothrock, S. Todorovic and S.C. Zhu, Joint Inference of Groups, Events and Human Roles in Aerial Videos, Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Boston, June, 2015.
[45]. Y. Zhu, Y.B. Zhao and S.C. Zhu, Understanding Tools: Task-Oriented Object Modeling, Learning and Recognition, Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Boston, June, 2015.
[46]. B. Zheng, Y. Zhao, J. Yu, K. Ikeuchi, and S.C. Zhu, Scene Understanding by Reasoning Stability and Safety, Int'l Journal of Computer Vision, vol. 112, no. 2, pp221-238, 2015.
[47]. M. R. Amer, S. Yousefi, R. Raich, and S. Todorovic, Monocular Extraction of 2.1D Sketch Using Constrained Convex Optimization, Int'l J. of Computer Vision, 2015 .
[48]. A. Barbu, T.F. Wu and Y. N. Wu, Learning Mixtures of Bernoulli Templates by Two-Round EM with Performance Guarantee, Electronic Journal of Statistics, 2015.
[49]. C.M. Xiong, N. Shukla, W. Xiong and S.C. Zhu, Robot Learning with a Spatial, Temporal, and Causal And-Or, Int'l Conference on Robotics and Automation (ICRA), 2016.
[50]. Y. Lu, S.C. Zhu and Y.N. Wu, Learning FRAME Models Using CNN Filters, 30th AAAI Conference on Artificial Intelligence (AAAI), Phoenix, Arizona, 2016.
[51]. C.S. Liu, J. Y. Chai, N. Shukla and S.C.Zhu, Task Learning through Visual Demonstration and Situated Dialogue, AAAI Workshop on Symbiotic Cognitive Systems, Phoenix, Arizona, 2016.
[52]. S. Park and S.C. Zhu, Attributed Grammars for Joint Estimation of Human Attributes, Parts and Poses, Proc. of International Conference on Computer vision (ICCV), 2015.
[53]. Q.S. Zhang, Y.N. Wu and S.C. Zhu, Mining And-Or Graphs for Graph Matching and Object Discovery, Proc. of International Conference on Computer vision (ICCV), 2015.
[54]. T.F. Wu, B. Li and S.C. Zhu, Learning And-Or Models to Represent Context and Occlusion for Car Detection and Viewpoint Estimation, IEEE Trans on Pattern Analysis and Machine Intelligence, 2015